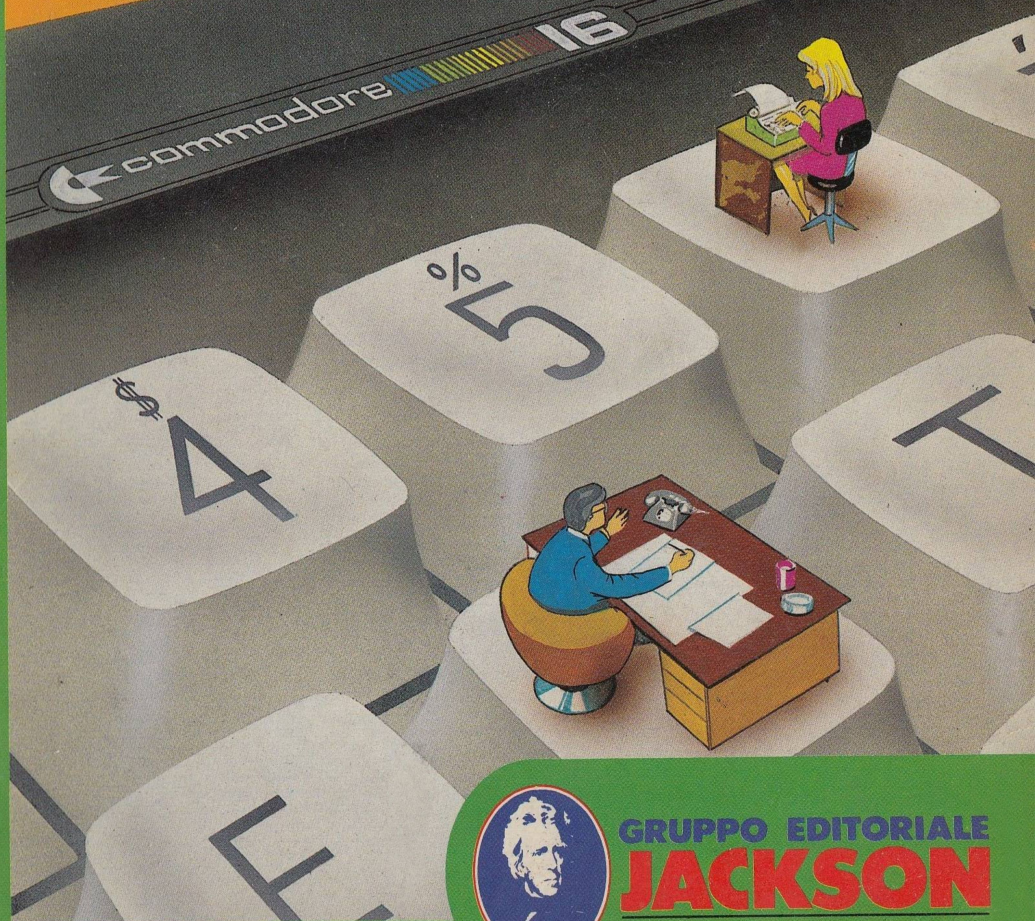
Alessandro Borra Mauro Cristuib Grizzi

Lavoriamo con il COMMODORE 16

HOME E PERSONAL COMPUTER

Lavoriamo con il COMMODORE 16

Alessandro Borra
Mauro Cristuib Grizzi

GRUPPO
EDITORIALE
JACKSON
Via Rosellini, 12
20124 Milano

SOMMARIO

INTRODUZIONE

Fin dai tempi più remoti, il problema del calcolo si è dimostrato essere per l'uomo uno dei più difficili da risolvere. Di pari passo con la propria evoluzione, l'umanità si è posta domande di carattere quantitativo via via più complesse alle quali non riusciva a dare risposte immediate: quando è il periodo migliore per la semina, quanto distano tra loro due luoghi, quale può essere il valore di un dato oggetto o di una prestazione, quali leggi regolano il moto degli astri nel cielo. Quasi subito si accorse della incapacità di affrontare questi quesiti con il solo uso delle dieci dita: naquero così i primi strumenti di calcolo. L'abaco, una tavoletta tracciata a righe parallele (simile all'odierno pallottoliere), compare nel 3000 a.C. presso i babilonesi. Questo primitivo strumento di calcolo manuale dava però ai sacerdoti egizi la possibilità di determinare con assoluta precisione le entità e i confini dei singoli appezzamenti attorno alla foce del Nilo, in modo da tornare ai legittimi proprietari dopo l'annuale piena del fiume la medesima terra che avevano prima, senza sbagliare di un centimetro. Era un problema complesso, e forse per la prima volta un problema di tipo numerico veniva ad assumere un'importanza addirittura vitale per la società.
Da allora passarono molte piene del Nilo, e l'uomo sempre più si trovò immerso nei problemi di calcolo: un problema poteva essere facile nella determinazione del processo di risoluzione, ma poteva diventare irrealizzabile la sua soluzione se i conti, anche se banali, fossero stati molti. Un uomo sa svolgere senza difficoltà una addizione tra due numeri, ma non può assolutamente sperare di risolverne centomila: ecco un problema dal banale processo di risoluzione che però nessuno può affrontare senza un adeguato strumento di calcolo.
Il 1642 rappresenta una data miliare nella storia della computazione: il diciannovenne matematico francese Blaise Pascal costruisce la prima addizionatrice meccanica: la 'Pascalina'. Per la prima volta veniva realizzato il riporto automatico: per oltre trecento anni esso fu il principio fondamentale per tutti gli strumenti di calcolo, dal contachilometri delle automobili alle macchine calcolatrici. Ma questa data ha un'altra importanza fondamentale: per la prima volta nella storia dell'uomo una macchina poteva eseguire dei conti in modo autonomo, senza l'intervento umano, anzi l'utilizzatore poteva addirittura ignorare completamente il modo di

risolvere un'addizione! E così, per ironia della sorte, mentre l'umanità cercava di togliersi di dosso il problema del calcolo, un nuovo problema, tuttora insoluto, veniva a porsi: quello del controllo. Un uomo *sa* se un proprio conto è giusto, ma come fidarsi della correttezza del risultato fornito da una macchina, che non ha la coscienza di ciò che sta facendo? Un sistema naturalmente c'è: rifare i conti, e così... rieccoci da capo!
Il problema del controllo è però cosa recente, la storia della computazione passa ancora da tante, storiche date. 1671: il tedesco Leibniz costruisce una macchina che esegue anche le moltiplicazioni e le divisioni. Purtroppo bisogna attendere la rivoluzione industriale perchè questi e altri prototipi vengano realizzati su grande scala e commercializzati: l'espansione del commercio e delle banche coprì tutte le forniture di macchine di calcolo automatiche. 1890: il Governo degli Stati Uniti (i quali contano già 63 milioni di abitanti) decide di affidare a Herman Hollerit la gestione del censimento nazionale: in soli due anni e mezzo sono disponibili i dati finali, grazie ad una particolare macchina elettrica che utilizza come dati dei fori su schede di cartoncino dalle dimensioni di una banconota da un dollaro... siamo alle soglie dell'odierno computer: l'uomo ha ormai trovato la strada che lo porterà alla soluzione del problema del calcolo (con la conseguenza della perdita di affidabilità), e d'ora in avanti sarà quasi solo un problema di tecnologia.
L'anello di congiunzione (chiamiamolo così) tra le macchine di calcolo e l'odierno computer è la macchina che nel 1943 l'Università di Pennsylvania propone all'esercito statunitense per risolvere ad alta velocità i problemi balistici dell'artiglieria. Siamo in tempo di guerra, e i militari accettano: in due anni viene realizzato ENIAC (Integratore ed Elaboratore Numerico Elettronico): pesante 30 tonnellate, occupava una superficie di 180 metri quadrati e poteva eseguire 300 moltiplicazioni al secondo. Parallelamente (nel 1945) lo scienziato di origine ungherese John von Neumann getta le basi del moderno computer: studia e poi realizza una nuova macchina che possiede una memoria nella quale trovano posto contemporaneamente le istruzioni di funzionamento e i dati di utilizzo. A tutt'oggi questa struttura resiste ancora, e von Neumann viene universalmente considerato il 'padre' del moderno calcolatore. Nel 1952 l'Università di Princeton realizza EDVAC (Calcolatore Automatico Elettrico a Quantità Discrete), sulle basi delle teorie dello studioso ungherese.

La macchina di von Neumann, l'odierno computer, si dimostra essere subito versatilissima, dai molteplici campi di applicazione, anche se, storicamente, deve la sua creazione al problema del calcolo e del conteggio (tanto che ancora oggi i francesi la chiamano *ordinateur*). L'uomo è finalmente riuscito a ridurre il proprio lavoro e a migliorarne la qualità. La fisica, la matematica e l'astronomia hanno avuto a disposizione un potente strumento, che ha portato grossi ritorni nella vita di tutti i giorni. Ma la storia delle origini del computer non è ancora finita: mancano ancora le date che si riferiscono agli ultimi trenta frenetici anni.
Gli anni '50 si aprono con le cosiddette 'macchine della prima generazione', costituite da tubi a vuoto e capaci di 2200 moltiplicazioni al secondo, e si chiudono

con quelle della seconda: transistor e 38000 operazioni al secondo. I tempi si accorciano e i computer della terza (e attualmente ultima) generazione compaiono nell'aprile 1964 (Sistema IBM 360): sono costituiti da circuiti integrati monolitici, il silicio ormai diventa l'elemento fondamentale. Ora si parla di due milioni di moltiplicazioni al secondo, e l'unità di tempo diventa il nanosecondo (miliardesimo di secondo). Per avere un'idea delle grandezze di cui stiamo parlando, si sappia che un nanosecondo sta ad un secondo come questo sta a 30 anni! Inoltre in un nanosecondo la luce, che viaggia alla velocità di trecentomila chilometri al secondo, percorre appena 30 centimetri.
Fino ad ora, però, il computer è ancora voluminoso e soprattutto costoso, e trova applicazione per lo più in ambienti scientifici o militari, ma l'avvento dei voli spaziali segnerà l'inizio della sua odierna diffusione capillare. Infatti il problema dei volumi e dei pesi, di primaria importanza in campo aerospaziale, 'costrinse' l'industria elettronica statunitense ad un grosso sforzo: nel 1971 la Intel presenta il primo microprocessore. Da allora il crollo dei prezzi e l'aumento delle prestazioni è stato vertiginoso: basti pensare che solo una decina di anni fa (nel 1974) la Texas Instruments brevettava la prima calcolatrice elettronica tascabile, dotata delle sole quattro operazioni fondamentali. Oggi il personal computer è arrivato alla portata di chiunque.
Questa breve introduzione storica è necessaria per capire appieno l'evoluzione del computer nel corso dei secoli e la sua funzione nell'odierna società. Abbiamo voluto spiegare in questo modo come mai il presente libro si propone di *lavorare* con esso. Quella descritta è la testimonianza di una vera e propria rivoluzione: il calcolatore, concepito inizialmente come strumento di puro calcolo, si è dimostrato in realtà capace di elevare la qualità del lavoro dell'uomo, aumentandone la produttività. Uscito dai laboratori scientifici, i campi nei quali ha saputo mostrare la sua utilità sono stati innumerevoli: negli uffici, nelle industrie e negli ambienti di ricerca il computer si è ovunque dimostrato un valido supporto al lavoro dell'uomo. È in quest'ottica che bisogna considerare il calcolatore: è uno strumento che permette di lavorare meglio, è in fondo come una penna stilografica, ma dalla potenza enormemente maggiore.
Ora il computer ha fatto il suo ingresso nelle case di ognuno, grazie a quel 'fratello minore' che è lo 'home computer'. Voi stessi, con l'acquisto del Commodore 16, siete entrati nel mondo informatico: il futuro è di questo strumento, ed è giusto capirne da subito le possibilità. Data la sua grande versatilità, lo 'home' è stato presentato e lanciato, grazie ai suoi colori, suoni e giochi, come uno strumento di distrazione. Ma ora voi sapete che un computer non è solo questo, e il presente libro vuole mostrare alcune delle applicazioni più importanti ed utili nella vita quotidiana:

— **wordprocessor**. Per scoprire la differenza tra il battere a macchina e l'utilizzare il C16 per scrivere, manipolare (nel più vasto senso del termine) e memorizzare testi anche lunghissimi. Il tempo è prezioso, e va utilizzato con intelligenza: questo programma interamente in linguaggio macchina vi permetterà un uso realmente professionale del calcolatore;

— **data base**. Un potente strumento per organizzare in modo naturale e veloce ogni tipo di informazione: dalla rubrica telefonica all'agenda, dalla biblioteca personale all'archivio di qualunque genere. Scoprirete il vantaggio dell'utilizzo del calcolatore rispetto alla semplice carta. Il programma è presentato in due versioni: per l'uso di un'unità a dischi, dall'incredibile velocità di ricerca, e per il registratore a cassette, versatile ed efficace;

— **spreadsheet**. Insieme al word processing, il tabellone elettrico è l'applicazione più diffusa del personal computer negli ambienti di lavoro. Correla tra loro funzioni e dati numerici, rendendo automatico il calcolo in modo da permettere di analizzare un bilancio o di compiere previsioni di qualunque tipo. Il computer è stato costruito principalmente per calcolare, e vi accorgerete della naturalezza e semplicità con cui il vostro C16 vi rivelerà questo aspetto della sua poliedrica personalità.

Con i programmi di questo libro abbiamo voluto in qualche modo dare una panoramica, quanto più professionale possibile, dell'affascinante mondo del calcolatore, cercando di avviare il lettore-utente alla comprensione e alla compartecipazione di quella 'cultura informatica' che ogni giorno di più modifica il nostro modo di vivere e di pensare.

CODIFICA DEI PROGRAMMI

I listati dei programmi BASIC presenti nel libro sono stati realizzati mediante un'opportuna codifica in modo da renderli più leggibili, sostituendo dei mnemonici opportuni ai caratteri di controllo. Qui sotto è presentata la tabella di conversione:

```
{HOME}.........HOME
{CLR}..........PULIZIA SCHERMO
{CUR.SU}.......CURSORE IN ALTO
{CUR.GIU}......CURSORE IN BASSO
{CUR.DES}......CURSORE A DESTRA
{CUR.SIN}......CURSORE A SINISTRA
{SPC}..........SPAZIO
{RVS ON}.......REVERSE ON
{RVS OFF}......REVERSE OFF
{FLASH ON}.....FLASH ON
{FLASH OFF}....FLASH OFF
{INST}.........INSERT
{BLACK}........COL. NERO (CTRL+1)
{WHITE}........COL. BIANCO (CTRL+2)
{RED}..........COL. ROSSO (CTRL+3)
{CYAN}.........COL. CIANO (CTRL+4)
{PURPLE}.......COL. PORPORA (CTRL+5)
{GREEN}........COL. VERDE (CTRL+6)
{BLUE}.........COL. BLU (CTRL+7)
{YELLOW}.......COL. GIALLO (CTRL+8)
{ORANGE}.......COL. ARANCIO (CBM+1)
{BROWN}........COL. MARRONE (CBM+2)
{YL-GREEN}.....COL. GIALLO-VERDE (CBM+3)
{PINK}.........COL. ROSA (CBM+4)
{BL-GRN}.......COL. BLU-VERDE (CBM+5)
{LT.BLUE}......COL. BLU CHIARO (CBM+6)
{D.BLUE}.......COL. BLU SCURO (CBM+7)
{LT.GREEN}.....COL. VERDE CHIARO (CBM+8)
```

Inoltre i caratteri grafici, ottenuti con la pressione dei tasti 'SHIFT' e 'CBM', sono codificati in modo da indicare il tasto da premere assieme a 'SHIFT' o 'CBM'. Per esempio il cuoricino è codificato con {SH S}. Un numero dentro le parentesi, esempio {2 CUR.SU}, indica le volte che il tasto va premuto.

CAPITOLO 1

IL COMPUTER E IL WORDPROCESSING

1.1 Cos'è un wordprocessor

Una delle più tipiche applicazioni del computer in un ambiente di lavoro è sicuramente il 'wordprocessing', l'elaborazione, cioè, di testi. Il vantaggio che dà un tale programma rispetto ad una semplice macchina da scrivere è enorme: il testo, lettera o altro, può essere infatti digitato per intero anche se molto lungo, gli errori di sintassi possono essere eliminati, parole o interi paragrafi possono essere spostati e tutto ciò prima ancora di effettuare la stampa! Inoltre la possibilità di memorizzare su opportuno supporto fisso (in genere dischi o nastri) il proprio documento per poterlo in seguito rivedere o modificare rende un wordprocessing uno strumento di lavoro del tutto insostituibile.
Per chi studia, certamente un potente wordprocessor è quanto di meglio si possa avere, non solo per mettere in ordine gli appunti, ma soprattutto per lavori di ricerca: d'ora in avanti non esisterà più il doppio lavoro della 'brutta' e della 'bella' copia. Infatti il grosso vantaggio del wordprocessor consiste proprio nel poter elaborare contemporaneamente sia il progetto che la sua stesura finale, proprio perché, istante per istante, le si hanno entrambe sotto completo controllo.
Naturalmente un wordprocessor ha bisogno alla fine di una stampante, in modo da poter ottenere su carta il proprio scritto, ma si può fare a meno di essa nella fase di stesura, in quanto è possibile vedere, direttamente sullo schermo, come verrà la stampa e quindi intervenire sulla formattazione dello scritto finale prima ancora di averlo ottenuto! È quindi possibile lavorare con un wordprocessor anche senza la stampante: è sufficiente infatti memorizzare il proprio lavoro su un dischetto o una cassetta ed effettuare la stampa in un secondo momento.
Vi accorgerete dei grandi vantaggi che offre un wordprocessor solo con l'uso: dopo non potrete più farne a meno. EASYWORD, il programma che vi viene presentato in questo capitolo e che si trova nella cassetta allegata, è inoltre completamente in linguaggio macchina, in modo da essere potente e soprattutto veloce: un efficace strumento di lavoro deve essere 'trasparente' all'utente, nel senso che non deve rallentarlo o in qualche modo disturbarlo nell'esercizio delle proprie mansioni. Infine la sua semplicità d'uso permetterà una rapida familiarità anche a chi è alle prime esperienze con il calcolatore.

1.2 Cosa deve fare un wordprocessor

Vediamo ora in breve quali sono le caratteristiche fondamentali che deve avere in generale un wordprocessor (e in particolare vedremo come sono state realizzate in EASYWORD), in modo da capirne in pieno le funzioni.
Innanzitutto deve poter essere possibile digitare direttamente il testo oppure caricarlo da una periferica (a dischi o a nastri indifferentemente), ed è utile la possibilità di non cancellare con il caricamento il testo eventualmente già presente in memoria, in modo da poter unire due testi differenti.
Devono essere possibili il salvataggio dello scritto sulla periferica selezionata e la sua cancellazione dalla memoria, in modo da passare al termine di un lavoro al successivo senza avere il vecchio documento ancora sotto gli occhi.
Per quanto riguarda le funzioni di *editing*, cioè quelle che permettono di scrivere, modificare e cancellare il testo, le richieste sono molte e precise. Prima di tutto le azioni che il computer permette in ambiente BASIC (cioè al momento dell'accensione della macchina) devono essere mantenute inalterate, in modo da facilitare il processo di ambientamento al nuovo prodotto, quindi deve poter essere possibile inserire e cancellare caratteri mediante il tasto INST/DEL (con SHIFT si inserisce e in modo diretto si cancella). Il tasto CLR/HOME deve mantenere la stessa funzione cui si è abituati: premuto da solo deve posizionare il cursore in alto a sinistra nello schermo senza cancellare il testo presente sul video, premuto invece unitamente a SHIFT deve permettere la cancellazione dell'intero documento. Ma vediamo in particolare quali sono le richieste che generalmente deve soddisfare un wordprocessor riguardo le funzioni di *editing*:

— movimento del cursore. Il cursore (che indica il punto all'interno del testo dove apparirà il successivo carattere premuto sulla tastiera) deve poter essere spostato liberamente e facilmente all'interno dello scritto: in particolare è comodo poterlo muovere di un carattere, di una parola o di una frase per volta. Deve poter essere possibile anche lo spostamento immediato a fine e a inizio testo.

— devono essere implementati due modi operativi: il modo normale ed il modo inserimento. Nel secondo caso, tutto il testo successivo al cursore viene automaticamente spostato a sinistra in modo da impedire che le nuove parole cancellino quelle già presenti sul video (in pratica è ciò che succede quando, digitando un programma BASIC, si premono i tasti 'ESC' e 'A'). In questo modo per inserire ad esempio la parola 'bella' in 'la casa' per ottenere 'la bella casa' è sufficiente posizionarsi dopo l'articolo, entrare in modo inserimento e quindi scrivere ciò che si vuole. Comodo e facile da usare.

— deve essere presente una funzione di 'ricerca'. Si intende qui la possibilità di cercare, in modo automatico, una o più parole all'interno del testo senza doverlo leggere tutto. Inoltre deve esistere la possibilità di modificare o meno tale parola con una precedentemente definita, oppure decidere di volta in

volta se sia o meno il caso di operare la modifica. Un esempio chiarificatore può essere il seguente: supponiamo di aver scritto più volte nel testo la parola 'eccezzionale', che (mea culpa!) va scritta con un'unica zeta. Quando ci si accorge dell'errore, è sufficiente andare a cercare tutte le doppie zeta ('zz'), predisponendo come sostituzione una sola zeta ('z'): si aziona quindi la ricerca automatica ed ogni volta che si capita sulle due zeta della parola incriminata si opera la sostituzione (che viene eseguita dal computer alla pressione di un solo tasto!), mentre in tutti gli altri casi si continua la ricerca fino al termine del testo. È chiaro che in questo modo si è assolutamente sicuri di aver corretto ovunque questo errore, e di evitare quindi una pessima figura!

— cancellazione di parte del testo. Naturalmente ogni wordprocessor ha un proprio protocollo di cancellazione, ma quello realizzato in EASYWORD è estremamente potente e facile da usare: pochi altri programmi, anche per macchine più potenti quali l'Apple II o il Commodore 64, possono vantarsi di offrire le stesse prestazioni. In pratica è possibile cancellare del testo, partendo dal cursore: si può scegliere se cancellare un'unica parola, una frase (fino al primo punto: fermo, esclamativo o interrogativo che sia) o un intero paragrafo (cioè fino a quando non si va a capo di una nuova riga). L'unicità consiste nel poter eliminare sia il testo che segue il cursore che quello che lo precede! Questa duplice possibilità, una volta raggiunta l'opportuna familiarità, si rivelerà più che utile. Però la caratteristica principale dell'operazione di cancellazione offerta da EASYWORD è la possibilità di poter recuperare il testo appena cancellato, in modo da rimediare ad errori che in quasi tutti gli altri programmi sono fatali.

— spostamento di parti di testo. Spesso, durante la stesura di uno scritto, capita di voler spostare di posto una frase o un periodo per svariati motivi: comprensione, ripetizione o... stile. Utilizzando carta e penna gli asterischi, i rimandi e le frecce si sprecano, e tutto ciò a discapito della leggibilità, che in questo caso diventa frammentata; non è raro poi il caso in cui ci si accorga di difficoltà di lettura solo dopo essere passati alla copia finale. Tutto ciò non accadrà più con l'utilizzo del wordprocessor: è infatti possibile spostare (e addirittura duplicare!) parole, frasi o interi paragrafi solo con la pressione di pochi tasti; e se poi ancora non si è soddisfatti non c'è che da continuare: è facile risolvere questo tipo di problema quando si ha sempre sottomano l'ultima versione del proprio scritto.

Occupiamoci ora di uno degli aspetti più importanti di un wordprocessor: la stampa su carta del testo presente in memoria. Esistono in proposito due tipi di programmi: quelli per cui ciò che appare sul video è esattamente quello che si vedrà sulla carta e quelli per cui invece ciò non accade. Diciamo subito che il Commodore 16, potendo contenere sul video soltanto 40 caratteri per riga, non permette di avere sempre sott'occhi lo scritto così come apparirà in stampa finale

(a meno di limitarsi a pagine tanto strette da poter essere contenute sullo schermo, ma in questo caso quanto può essere utile un simile wordprocessor?), e quindi bisognerebbe considerare il video come una 'finestra' sul testo, con la possibilità di movimento nelle quattro direzioni. Questa tecnica è però scomoda in fase di battitura: infatti in questo modo non si può avere sempre sotto controllo tutto il testo, ma solo una sua parte per volta, a tutto discapito della leggibilità, e sappiamo quanto sia fondamentale in questa fase la rilettura del proprio scritto. Solo computer con output a video più esteso, quali ad esempio l'Apple II, permettono wordprocessor del primo tipo.
Il programma EASYWORD appartiene alla seconda classe di programmi, ma permette comunque di vedere sul video il proprio scritto: infatti è possibile selezionare la periferica di output, e quindi scegliere lo schermo televisivo. È inoltre possibile selezionare a proprio piacere, nella fase di battitura, tutti i fondamentali parametri di stampa (più qualche altro che vedremo in seguito) quali il numero della colonna a sinistra e a destra entro le quali starà lo scritto sul foglio, il numero di righe stampate per pagina o la lunghezza, in righe, del foglio di carta utilizzato (i valori assunti inizialmente da EASYWORD per questi parametri sono rispettivamente 5, 75, 66 e 72). Come detto, tali quantità (che hanno un valore di *deafult*, cioè che assumono in caso l'utente non li definisca) possono essere definite durante la fase di battitura del testo: per fare ciò è necessario definire dei caratteri particolari (o codici di formato) che permettano di distinguere il testo vero e proprio dai comandi di stampa. La distinzione consiste nel fatto che i comandi di formato appaiono sullo schermo in *reverse*, e un numero successivo definisce il nuovo valore che dovrà essere tenuto in considerazione per quel parametro in fase di output, evitandone la stampa. Per esempio il codice per la definizione del margine sinistro è una 'l' in campo inverso, e la decisione di far partire il testo dalla settima colonna sarà realizzata da:

l7

Le possibilità offerte da EASYWORD in fase di stampa sono molteplici: centratura di parole o frasi, sottolineatura (naturalmente solo per le stampanti per cui ciò sia possibile), numerazione automatica delle pagine, possibilità di stampare su ogni pagina un *header* e un *footer* (cioè delle scritte sopra e sotto il testo, ad esempio per richiamare il nome del capitolo cui appartiene la pagina o per utilizzare la numerazione romana per le pagine della prefazione).
Infine un'ultima importante particolarità: la memoria a disposizione per il testo è naturalmente limitata, e ciò perchè nei 12 Kbyte di RAM libera del Commodore 16 devono trovare posto, oltre al testo, anche le istruzioni di EASYWORD e le locazioni che esso utilizza per il proprio funzionamento; quindi è stata realizzata un'opportuna funzione di *link* che permette di connettere tra loro, durante la procedura di stampa, testi precedentemente memorizzati su disco o nastro. In questo modo è possibile ottenere la stampa continua anche di un documento molto più lungo di quello che può trovare posto nella memoria del computer, senza che appaiano antipatiche interruzioni sulla carta.

1.3 I comandi di EASYWORD

Dopo aver descritto le principali caratteristiche di un generico wordprocessor, con qualche accenno a EASYWORD, questo paragrafo vuole ora trattare approfonditamente tutti i comandi (le direttive per la formattazione della stampa sono invece trattate nel paragrafo 1.4) del potente elaboratore di testi presentato in questo libro, che, per onor di precisione, è la conversione, opportunatamente addattata e modificata, del programma 'Speedscript 3.0', apparso sulla rivista Supercommodore, edita da J. Soft.
Consigliamo, le prime volte, di leggere queste righe di fronte al computer dopo aver caricato in memoria EASYWORD, in modo da avere un immediato riscontro pratico alle specifiche date in questo paragrafo. In fondo al capitolo, dopo il commento al listato del programma, è presente uno schema riassuntivo di tutti i comandi accettati da EASYWORD. Molti di essi sono realizzati con la pressione contemporanea dei tasti 'CTRL' e/o 'SHIFT'; per segnalare ciò adottiamo la seguente convenzione di scrittura (sia 'X' il tasto che esegue un determinato comando):

X il comando viene attivato dalla pressione del solo tasto 'X'
SH-X bisogna premere il tasto 'SHIFT' assieme ad 'X'
CTRL-X il tasto da premere contemporaneamente a 'X' è 'CTRL'
SH-CTRL-X sia 'SHIFT' che 'CTRL' vanno premuti assieme ad 'X'

EASYWORD accetta 43 comandi diversi: si è cercato per quanto possibile di assegnargli dei tasti particolari, in modo che fosse facile la memorizzazione della loro posizione. I comandi possono essere suddivisi in quattro classi:

— comandi di utility

CTRL-B ('B' sta per 'bordo') per cambiare il colore dello sfondo dello schermo;
CTRL-L ('L' sta per 'lettere') per cambiare il colore del testo;
CTRL-7 per conoscere quanti byte liberi si hanno ancora a disposizione per la scrittura del testo;
CTRL-P ('P' sta per 'pulizia') per svuotare il *buffer*: come spiegato in seguito il *buffer* permette la cancellazione e lo spostamento di parti di testo;
CTRL-D ('D' sta per 'dischetto') per poter dare un comando al *drive*;
CTRL-4 ('4' è il tasto su cui si trova '$', simbolo di *directory*) per ottenere la lista dei *file* presenti su dischetto.
Naturalmente questi ultimi due comandi sono utilizzabili solamente se è collegata un'unità a dischi.

— comandi di input/output:

HELP per caricare un *file* testo da nastro o da disco. In effetti, per aumentare la facilità d'uso di EASYWORD, i sette tasti funzione più il tasto 'HELP' sono utilizzati per realizzare altrettanti comandi;
F7 per memorizzare il documento presente in memoria sull'opportuna periferica;
CTRL-V('V' sta per 'verifica') per controllare la correttezza dell'operazione di memorizzazione;
CTRL-F('F' sta per 'formato') o **CTRL-K** per definire un codice di formato;
CTRL-O('O' sta per 'output', uscita dati) per ottenere la stampa del testo presente in memoria;
SH-CTRL-O per definire la periferica di output: in questo modo è possibile vedere su schermo come verrà stampato il documento.

— comandi di movimento del cursore:

HOME per posizionare il cursore in alto a sinistra senza cancellare il testo. Una seconda volta permette di andare all'inizio del testo;
CLR per cancellare l'intero documento presente in memoria;
CTRL-Z (la zeta, come ultima lettera dell'alfabeto, simboleggia la fine) per posizionare il cursore al termine del testo;
CURSORE A DESTRA o **CTRL-;** per spostare il cursore di un carattere a destra;
CURSORE A SINISTRA per spostare il cursore di un carattere a sinistra;
F1 per spostare il cursore di una parola a destra;
F4 per spostare il cursore di una parola a sinistra;
F2 o **CURSORE IN BASSO** o **CTRL-Q** per spostare il cursore al periodo successivo;
F5 o **CURSORE IN ALTO** per spostare il cursore al periodo precedente;
F3 per portare il cursore al paragrafo successivo;
F6 per portare il cursore al paragrafo precedente.

— comandi di editing

DEL per cancellare un carattere di testo prima del cursore;
INST per inserire uno spazio alla posizione del cursore;
CTRL-K ('K' sta per 'kill', uccidere, in modo da intendere l'effetto) per cancellare, carattere per carattere, il testo successivo al cursore;
SH-CTRL-K per cancellare tutti gli spazi successivi al cursore;
CTRL-G per tabulare, ossia per inserire 5 spazi bianchi nel testo;
SH-RETURN per inserire 2 simboli di 'RETURN' e 5 spazi vuoti;
RUN (testo RUN /STOP assieme a 'SHIFT') per inserire 255 spazi;
CTRL-X per invertire tra loro due caratteri;
CTRL-A per cambiare un carattere da maiuscolo in minuscolo e viceversa;
CTRL-I ('I' sta per 'inserimento') per passare dal modo normale al modo inserimento e viceversa;

CTRL-E ('E' sta per 'eliminazione') o **CTRL-2** per cancellare parti del documento dopo il cursore. La sua chiamata cancella automaticamente il *buffer*;

SH-CTRL-E o **SH-CTRL-2** idem di CTRL-E, solo che il *buffer* non viene cancellato;

CTRL-W per cancellare parti del testo prima del cursore (notate che, sulla tastiera la 'W' è a sinistra della 'E': la vicinanza e la posizione stanno a suggerire la funzione). Questo comando pulisce automaticamente il *buffer*;

SH-CTRL-W idem di CTRL-W, solo che il *buffer* non viene cancellato;

CTRL-R ('R' sta per 'richiamo') o **CTRL-9** per richiamare il contenuto del *buffer*;

CTRL-£ per effettuare automaticamente la ricerca e sostituzione di parti del testo;

SH-CTRL-H ('H' sta per 'hunt', cercare) per definire una parola (o più) da cercare all'interno del testo;

CTRL-H per effettuare la ricerca all'interno del testo;

SH-CTRL-J per definire una parola (o più) da sostituire. Va utilizzato solo unitamente al comando di ricerca: notare la vicinanza sulla tastiera tra la 'J' e la 'H';

CTRL-J per effettuare la modifica.

Come potete vedere i comandi a disposizione sono molti, e magari non di tutti si è capito il funzionamento. Il metodo migliore è quello di provarli direttamente alla tastiera del computer, comunque poco avanti tutti sono ampiamente commentati.
Dato il 'RUN' a EASYWORD, ci si accorge immediatamente che il video, nero con le scritte azzurro chiaro, appare diviso in due parti: una prima linea interamente in *reverse* e il resto completamente vuoto, con il cursore che lampeggia all'inizio della seconda riga. In realtà la prima linea è la FINESTRA DI COMANDO: non è possibile scriverci parte del testo; praticamente questa riga permette la comunicazione tra voi e EASYWORD: qui appariranno i vari messaggi che il programma fornisce di volta in volta e qui dovrete rispondere, quando richiesto, alle sue domande, per esempio specificando il nome del *file* di testo da caricare o da registrare o ancora le parole da cercare e sostituire all'interno del documento.
Nelle restanti 24 righe sarà invece possibile digitare il testo: lo *scroll* che si avrà al completamento della linea inferiore non coinvolgerà naturalmente la finestra di comando. Il movimento del cursore è vincolato dal testo, nel senso che non può essere spostato oltre l'ultima riga o dopo la freccia a sinistra, carattere che indica la pressione del tasto 'RETURN' e quindi la fine di un paragrafo.
È da evitare, prima di dare il 'RUN' al programma, l'utilizzo delle funzioni permesse dal tasto 'ESC' (per esempio 'ESC' più 'M', che disabilita lo *scroll*), pena comportamenti irregolari di EASYWORD.
Osserverete che, subito dopo il 'RUN' al programma, la linea di comando contiene il messaggio:

Easyword di A. Borra

Il nome del programma rimarrà sempre presente (a meno di essere in modo inserimento, nel quale caso un opportuno messaggio vi segnalerà il particolare modo operativo), mentre alla pressione di un qualsiasi tasto il nome dell'autore scomparirà, così da non essere costretti a 'subirne' la presenza per tutta la durata del lavoro: la discrezione innanzitutto!
Provate ora a digitare qualche parola: a fine linea osserverete l'effetto del *word-wrapping*, una particolare tecnica che permette di non spezzare le parole andando a capo di riga: se la parola che state battendo non sta interamente su un'unica linea, essa viene automaticamente spostata per intero sulla riga successiva, unitamente al cursore, e qui potrete completarla. Questo sistema, veloce e di nessun fastidio in fase di battitura, rende particolarmente leggibile sullo schermo il documento digitato: é questa un'altra particolarità (permessa solo dal linguaggio macchina) che rende EASYWORD uno dei migliori elaboratori di testo disponibili per home computer, anche se paragonato a quelli commericiali.
Occupiamoci ora dei vari comandi, descrivendone particolareggiatamente l'effetto di ognuno.

1.3.1 I comandi di utility

CTRL-B — Nella versione qui presentata, EASYWORD ha come colore dello sfondo il nero, e questo per il fatto che l'occhio umano compie uno sforzo minore se la luminosità è bassa: infatti tutti i video professionali hanno questa caratteristica. Però il programma offre la possibilità di modificare il colore dello sfondo (e contemporaneamente quello del bordo), in modo che possiate scegliere quello che ritenete essere il più comodo per voi. La luminosità è fissata al massimo (7), mentre la scelta può variare su 16 colori (vedere in proposito il manuale d'uso che accompagna la macchina al momento dell'acquisto).

CTRL-L — Il colore fissato per il testo da EASYWORD è l'azzurro chiaro, ma, analogamente a quello dello sfondo, potete cambiarlo a piacere. La scelta cade sempre sui 16 colori che il Commodore 16 mette a disposizione, mentre la luminosità è fissata a 4: come spiegato nel paragrafo 1.5.2, è possibile modificare quest'ultimo valore mediante una POKE in memoria prima di eseguire il programma. EASYWORD offre poi un'importante possibilità: i colori dello sfondo e del testo possono essere memorizzati in modo definitivo, in modo tale che ad ogni caricamento successivo il programma utilizzi automaticamente quelli da voi definiti. Per fare ciò, bisogna caricare il programma originale (quello cioè fornito nella cassetta allegata al libro), dare il 'RUN' e quindi utilizzare i comandi CTRL-B e CTRL-L fino a fissare i colori scelti, dopo, tenendo premuto il tasto RUN/STOP, si pigia il pulsante di *reset* sul lato destro del computer. Osserverete che 'uscirete' dal programma, e vi troverete in ambiente MONITOR, con i colori che la macchina assume all'accensione. Togliete adesso il dito dal pulsante RUN/STOP e pigiate 'X'

più 'RETURN' per tornare al 'BASIC': potete ora tranquillamente registrare il programma su dischetto o nastro: la prossima volta che lo caricherete e lo eseguirete, i colori di EASYWORD saranno quelli da voi scelti; una comodità in più sicuramente molto utile. Noterete che, tornando al BASIC dopo aver dato il *reset*, gli otto tasti funzione sulla destra non sono più attivi: niente paura, è stato EASYWORD a fare ciò in modo da poterli utilizzare per altrettanti comandi.

CTRL-7 — Questo comando permette di conoscere quanti byte liberi si hanno ancora a disposizione per il testo (vengono mostrati nella finestra di comando); all'inizio si hanno esattamente 3,5 Kbyte (3584 byte) disponibili: essi vi permetteranno di realizzare due o tre pagine di stampa, a seconda del formato prescelto. Osserverete come ogni carattere occupi un byte, così come il simbolo di fine paragrafo (la freccia a sinistra: si ottiene premendo 'RETURN'), mentre tutto lo spazio vuoto alla fine di ogni linea non ne occupi alcuno: in questo modo si è riusciti ad ottimizzare la memorizzazione del testo. Praticamente questo spazio bianco è come se non ci fosse: neanche il cursore ci può andare sopra. Il comando CTRL-7 è particolarmente utile quando si digita un documento lungo: a seconda della quantità di memoria ancora a disposizione si può scegliere se memorizzare quanto scritto (è bene lasciare sempre un po' di spazio in fondo per eventuali correzzioni) e ripulire quindi la memoria oppure continuare a inserire del testo. Ricordate che avete la possibilità di connettere tra loro testi su *file* diversi (ved. il paragrafo 1.4), ed è comodo abituarsi ad avere singole parti brevi, in modo da diminuire i tempi di caricamento e di ricerca di particolari brani.

CTRL-P — Questo comando serve per pulire il *buffer* svuotandolo del suo contenuto; un opportuno messaggio:

Buffer svuotato

segnala l'avvenuta esecuzione. Il *buffer* è una zona di memoria (255 byte) che EASYWORD utilizza per metterci i caratteri che vengono cancellati dai comandi CTRL-E (o SH-CTRL-E) e CTRL-W (o SH-CTRL-W). La sua utilità è duplice: permette infatti di recuperare delle parti di documento erroneamente cancellate, e inoltre consente lo sposamento di interi brani da una parte all'altra del testo. Entrambe queste funzioni sono realizzate mediante CTRL-R, spiegato più oltre.

CTRL-D — Questo comando, unitamente al successivo, è utilizzabile solo se al computer è collegata un'unità a dischi accesa. Permette di dare al *drive* una qualsiasi direttiva; EASYWORD chiede di specificare quale (mediante il messaggio

Comando?

nella prima linea dello schermo). I comandi che possono essere dati sono:

'RETURN' per sapere lo stato del disco
i + 'RETURN' per inizializzare il *drive*
v + 'RETURN' per validare un dischetto
n,nome + 'RETURN' per pulire la *directory* di un dischetto
n,nome,id + 'RETURN' per formattare un nuovo dischetto
r:nnomefile =vnomefile + 'RETURN' per cambiare il nome ad un *file*
s:nomefile + 'RETURN' per cancellare un *file*

I comandi sono comunque tutti quelli descritti sul manuale di utilizzo del *drive* fornito assieme alla periferica all'atto dell'acquisto.

CTRL-4 — Questo comando permette di vedere su schermo, senza cancellare il testo presente in memoria, la *directory* del dischetto presente nel *drive*. Lo schermo diventa nero e scompare la finestra di comando. La lista dei *file* può essere interrotta in qualsiasi momento con la pressione dello spazio: riprenderà alla pressione di un qualsiasi tasto. Alla fine si può tornare al modo testo premendo 'RETURN'. In caso il *drive* sia spento o non sia collegato appare subito la richiesta

Premi RETURN

per tornare al documento.

1.3.2 I comandi di input/output

HELP — Permette il caricamento di un file di testo da disco o da nastro. Se il cursore è posizionato all'inizio del testo, EASYWORD cancella quello attualmente presente in memoria, altrimenti viene effettuato un *merge*: il nuovo testo viene caricato in coda a quello sulla macchina, a partire dalla posizione del cursore. In quest'ultimo caso fate attenzione a non andare oltre la memoria testo disponibile: il programma poi non funziona più regolarmente poichè in fondo alla memoria disponibile per il documento sono situate alcune variabili fondamentali. Dopo aver dato il nome del *file* alla domanda

Load:

che appare nella finestra di comando (è essenziale darlo anche se si utilizza il registratore) bisogna specificare la periferica utilizzata, premendo 'N' o 'D' (con evidente significato) alla richiesta

Nastro o Disco?

Quindi lo schermo, a meno che non lo sia già, diventa nero e viene iniziata la procedura di caricamento: se tutto funziona regolarmente apparirà l'opportuno messaggio

Ok

altrimenti EASYWORD segnalerà il tipo di errore riscontrato.

F7 — Analogo al comando precedente è questo, che permette però di registrare su nastro o disco il contenuto della memoria per salvare un documento in modo permanente. EASYWORD pone la domanda

Save:

cui bisogna rispondere con il nome che si vuole dare al *file* che conterrà l'attuale testo in memoria. Dopo aver chiarito se il salvataggio deve avvenire su nastro o disco, lo schermo diventa nero e viene eseguito il comando. Nel caso in cui si utilizzi il dischetto, EASYWORD automaticamente aggiunge i caratteri '0:' al nome del file, in modo da rendere possibile il salvataggio sul *drive* 0: naturalmente la cosa è utile solo nel caso di avere a disposizione un doppio *drive*. Infine se il nome contiene come primo carattere la chiocciola ('@'), che permette di cancellare il *file* su dischetto avente lo stesso nome e di registrarci 'sopra' il nuovo, il programma non aggiunge alcun carattere. I possessori di *drive* noteranno che i *file* di testo salvati sono di tipo PRG, come i normali programmi BASIC, ma ciò non ne permette il caricamento fuori da EASYWORD in quanto la loro struttura è completamente diversa.

CTRL-V — Questo comando permette di controllare un'operazione di registrazione appena avvenuta, ed è consigliata soprattutto con il registratore a cassette, essendo il lettore di dischi molto più affidabile. La procedura di utilizzo è del tutto analoga a quella dei precedenti due comandi: stavolta il messaggio che EASYWORD mostra per avere il nome del *file* da verificare è:

Verify:

Si ricorda che, in genere, per 'uscire' da una richiesta di EASYWORD è sufficiente premere il tasto 'RETURN'.

CTRL-F o **CTRL-=** — Questo comando permette di inserire all'interno del testo delle direttive per la formattazione della stampa. Queste, che verranno ampiamente descritte nel paragrafo 1.4, sono costituite in generale da un carattere in *reverse* e da un numero successivo. Per ottenere tali caratteri (che come detto non saranno stampati) bisogna utilizzare questo comando: EASYWORD chiederà con il messaggio

Tasto di formato:

quale codice deve essere inserito, e alla pressione di un qualsiasi tasto (incluso 'RETURN') il carattere corrispondente verrà posto nel testo alla posizione del cursore. Come spiegato meglio in seguito, non tutti i tasti attivano un comando di formato, e non tutte le posizioni all'interno del testo sono corrette per posizionarli.

CTRL-O — È il comando che permette di stampare su carta il documento presente in memoria, ed ha effetto soltanto se è connessa una stampante accesa. Prima di eseguire tale comando, è necessario posizionare la carta con correttezza, poiché EASYWORD, durante la stampa, tiene conto delle linee stampate in modo che lo scritto non vada a capitare sul tratteggio tra due fogli: la posizione corretta è con la testina scrivente della stampante proprio sopra la separazione tra due fogli. Nel paragrafo 1.4 vedremo come sarà possibile fare in modo che EASYWORD stampi correttamente con fogli di qualsiasi formato. La periferica selezionata da questo comando deve essere sul canale 4, e viene attivata con l'indirizzo secondario 7 (modo di stampa minuscolo/maiuscolo): tutte le stampanti Commodore possono essere attivate con questo comando; nel caso possediate una stampante diversa, la potete utilizzare con il prossimo comando.

SH-CTRL-O — In questo modo EASYWORD permette di scegliere il tipo di periferica di output mediante il messaggio su sfondo nero:

Stampa su: Video Disco Stampante?

a cui bisogna rispondere con la pressione dell'iniziale della periferica scelta. Rispondendo 'V' si potrà vedere sullo schermo come sarà la stampa su carta del nostro testo: l'unica differenza consiste nel fatto che il video ha solo 40 colonne, e quindi le righe più lunghe andranno a capo. Premendo il tasto 'SHIFT' avremo la sospensione dello scritto, fino a che non lo lasceremo; invece il tasto 'RUN/STOP' ha l'effetto di interrompere la stampa e di tornare al modo testo (ciò vale anche se si seleziona il disco o la stampante). Al completamento della stampa la pressione di qualsiasi tasto permette di tornare al modo testo. Rispondendo invece 'D' alla richiesta del tipo di periferica, la stampa avverrà su dischetto (dopo aver specificato il nome del *file*). L'utilità di questa opzione, a prima vista inutile, consiste nel fatto che il Commodore 16 è sprovvisto della possibilità di connettere stampanti via RS232: questo è un protocollo standard di comunicazione e fornisce la possibilità, qualora presente, di poter utilizzare qualunque tipo di periferica dotata del medesimo standard. In particolare le stampanti più prestigiose, quali ad esempio la Diablo a margherita, possono essere utilizzate solo via RS232. È però possibile servirsi ugualmente di tali stampanti se il testo è registrato per esempio su *drive* mediante un altro computer (ad esempio il Commodore 64) dotato di RS232, e da qui l'utilità della possibilità di stampare

su disco offerta da EASYWORD. Infine, scegliendo la stampante come periferica di output (tasto 'S'), si ha non solo la possibilità di definire il canale (non tutte le stampanti utilizzano il canale 4), ma anche l'indirizzo secondario, in modo da attivarla nel modo che si preferisce. Per quanto riguarda il valore del tale indirizzo, dipende dalla particolare stampante utilizzata: bisogna consultare in proposito il manuale che accompagna la periferica. Naturalmente fornire 4 alla domanda:

periferica?

e 7 a:

Indirizzo secondario?

equivale al comando CTRL-O.

1.3.3 I comandi di movimento del cursore

HOME — Analogamente al comportamento assunto in ambiente BASIC, la pressione del tasto CLEAR/HOME (senza SHIFT) permette di posizionare il cursore in alto a sinistra, senza cancellare il testo. Nel caso invece che il cursore sia già in quella posizione e il testo sia naturalmente sufficientemente lungo, il cursore si posizionerà automaticamente anche al suo inizio. Quindi per andare all'inizio del documento è sufficiente utilizzare due volte questo comando.

CLR — Permette la cancellazione dell'intero testo, dopo però aver dato la conferma alla domanda

ELIMINA TUTTO: Sei sicuro (S/N)?

Solo la pressione del tasto 'S' realizza la cancellazione totale del documento presente in memoria. Il comando deve essere eseguito con attenzione perché una volta cancellato, il testo non è più recuperabile: viene svuotato anche il *buffer*.

CTRL-Z — permette di posizionare il cursore alla fine del testo. Comando particolarmente comodo quando il testo è molto lungo.

CURSORE A DESTRA o **CTRL-;** — Questo comando, similarmente al BASIC, consente di spostare il cursore di un carattere a destra. A fine linea, il *word-wrapping* fa andare il cursore automaticamente all'inizio della successiva. Notate come non sia possibile oltrepasare la fine del testo o andare a fine riga dove non si trovano parole.

CURSORE A SINISTRA — In opposizione al precedente, questo comando ne realizza l'effetto opposto, spostando il cursore di un carattere a sinistra.

F1 — Consente di spostare il cursore dall'inizio di una parola alla successiva (sempre che ce ne sia una), velocizzando il suo movimento. Anche a questo comando non è consentito oltrepassare la fine del testo.

F4 — Esegue esattamente l'effetto opposto del precedente comando, portando il cursore all'inizio della parola precedente a quella su cui si trovava.

F2 o **CURSORE IN BASSO** o **CTRL-Q** — Questo comando permette di spostare il cursore all'inizio della frase successiva a quella su cui è il cursore. In pratica permette di andare al primo carattere che segue un punto fermo, un punto interrogativo, un punto esclamativo o un codice di 'RETURN' (simboleggiato dalla freccia a sinistra).

F5 o **CURSORE IN ALTO** — E' il comando opposto al precedente: posiziona il cursore all'inizio della frase precedente al punto nel quale ci si trova.

F3 — Con questo comando è possibile posizionarsi all'inizio del paragrafo successivo, cioè portare il cursore al primo carattere dopo il primo simbolo di 'RETURN' trovato.

F6 — Comando opposto a quello appena descritto, permette di muovere il cursore all'inizio del paragrafo che precede quello nel quale si trova il cursore. Come potete vedere, il movimento del cursore all'interno del testo è estremamente completo e veloce, oltre che facile da imparare: basta poca pratica per impadronirsi perfettamente della tecnica di spostamento.

1.3.4 I comandi di editing

DEL — Come in ambiente BASIC, la pressione del tasto DEL permette di eliminare un carattere prima del cursore, con lo spostamento dell'intero testo successivo. Osservate come anche in questa circostanza il *word-wrapping* impedisca lo spezzamento delle parole.

INST — Sempre per mantenere lo stesso utilizzo del tasto INST/DEL cui siete abituati quando digitate un programma, questo comando permette l'inserimento di uno o più spazi bianchi nella posizione del cursore, con spostamento a destra dell'intero documento successivo. È utile per inserire pochi caratteri all'interno di una frase: per inserimenti di entità maggiore è infatti più comodo ricorrere al modo inserimento (ved. in seguito).

CTRL-K — Questo comando serve per cancellazioni rapide di testo successivo

al cursore, che viene letteralmente 'succhiato' da esso (quasi fosse un... buco nero!). Il documento successivo naturalmente viene richiamato verso sinistra. Se non c'è più testo successivo al cursore, il comando agisce esattamente come DEL.

SH-CTRL-K — Posizionando il cursore su uno spazio bianco, questo comando permette la cancellazione di esso e di tutti i successivi (al massimo 255), spostando a sinistra l'eventuale testo successivo fino a raggiungere il cursore. Questa possibilità è molto utile in congiunzione al comando RUN ('SHIFT' e RUN/STOP, spiegato poco sotto) poiché permette la cancellazione immediata di tutti quegli spazi in più inseriti da RUN e non utilizzati per aggiunte di testo.

CTRL-G — È utilizzato per la tabulazione, nel senso che inserisce 5 spazi bianchi, eventualmente spostando il testo successivo. Il cursore alla fine si troverà dopo il quinto spazio inserito. Quando si scrivono delle lettere, all'inizio di ogni frase il testo è generalmente spostato verso destra (ved. figura 1), in modo da dare 'aria' allo scritto: a questo servono il comando appena visto e il successivo.

```
                                        Egr. Dott.
                                        Francesco Ramella
                                        Viale Donatello 12
                                        20136 MILANO



                                        Pontello, 13 settembre 1985

Egr. Dott. Francesco Ramella

     in riferimente alla Sua lettera del 4 settembre u.s. Le facciamo
avere le informazioni da Lei richieste sulla nostra localita' turistica
e le sue possibilita' alberghiere.

     Pontello, un piccolo ma moderno paese della Val Fontana a soli 5
chilometri dalla stazione scilstica di San Palumbo, presenta tutte
quelle caratteristiche che ne fanno un luogo di villeggiatura ideale
per ogni tipo di famiglia. L'imponente mole del Monte Grifone,
attorniato dalle altre vette alpine, rende una visione realmente unica
al mondo.

     E' presente ogni tipo di comodita', dal campo sportivo (tennis,
piscina, palestra e sauna) aperto a tutti i residenti al campo giochi
coperto per i piu' piccini, dai due cinema con proiezioni di film di
prima visione per i momenti di svago, fino ai caratteristici ristoranti
ove si possono gustare le caratteristiche pietanze regionali.

     La tranquillita' e il riposo sono...
```

Figura 1
Esempio di utilizzo dei comandi di tabulazione
CTRL-G e SH-RETURN

SH-RETURN — Questo comando inserisce due simboli di fine frase ('RETURN'), aggiungendo quindi 5 spazi bianchi. È da utilizzare alla fine di un paragrafo, perché in questo modo si va automaticamente a capo (primo 'RETURN'), si salta una riga (secondo 'RETURN') e si lascia dello spazio vuoto prima del paragrafo successivo (5 spazi bianchi).

RUN — Mediante la pressione contemporanea di 'SHIFT' e RUN/STOP è possibile inserire 255 spazi bianchi, con conseguente spostamento del testo successivo. In questo spazio generalmente trovano posto una o più frasi ed è quindi un comando utile per inserimenti all'interno dello scritto. Alla fine il comando SH-CTRL-K permette di eliminare gli spazi bianchi ancora presenti.

CTRL-X — Questo comando è una caratteristica peculiare di EASYWORD: permette di invertire tra loro il carattere su cui si trova il cursore con quello successivo, mentre il cursore non viene spostato. Ciò è utile quando, per fretta di battitura, due lettere vengono digitate nell'ordine inverso, esempio 'potrare' invece di 'portare': la correzione di questo tipo di errori è ora automatica in EASYWORD.

CTRL-A — Spesso, quando si sposta del testo, succede che frasi che erano all'inizio non lo sono più, e viceversa; in questo caso le maiuscole vanno convertite (e l'opposto, naturalmente). CTRL-A permette di realizzare ciò, cambiando il modo della lettera su cui si trova il cursore, che alla fine viene spostato di un posto a destra in modo da rendere velocissimo il cambiamento di un'intera parola o frase.

CTRL-I — Uno dei comandi più utili. Quando si vuole inserire del testo all'interno del documento i comandi INST, CTRL-G o RUN funzionano perfettamente, ma necessitano di tenere sempre d'occhio il video per evitare di andare inavvertitamente a scrivere sopra il testo già digitato. CTRL-I, invece, sposta automaticamente lo scritto successivo al cursore in modo da impedire antipatiche sovrapposizioni: questo modo di operare è ricordato continuamente dal messaggio

Modo inserimento

che appare nella finestra di comando al posto del solito 'Easyword', in modo da sapere sempre in quale stato ci si trova. Il comando praticamente funziona da 'interruttore', abilitando il modo inserimento quando ci si trova in modo normale, e viceversa; nel secondo caso appare il messaggio

Modo normale

e dalla pressione del primo tasto in poi torna a comparire in prima riga il nome del programma. In modo inserimento, tutti gli altri comandi funzionano come al solito, senza alcuna differenza.

CTRL-E o **CTRL-2** — Permette la cancellazione di testo successivo al cursore. Questo comando ha lo stesso effetto del successivo a parte il fatto che con CTRL-E il *buffer* viene automaticamente pulito alla chiamata, mentre con SH-CTRL-E questi mantiene il proprio contenuto precedente; vedremo in seguito quando può essere utile questa possibilità. Al comando, EASYWORD risponde con la seguente richiesta:

Elimina (V,F,P): RETURN esce

In pratica è possibile eliminare un vocabolo (pressione di 'V'), una frase ('F') o un paragrafo ('P'); la differenza tra una frase e un paragrafo è che la prima ha termine al punto (fermo, esclamativo o interrogativo) o al 'RETURN', mentre il secondo al solo 'RETURN'. Prendiamo per esempio la frase: 'Pippo accese la macchina. Ma questa non voleva sapere di mettersi in moto.←' (← è il simbolo di 'RETURN' utilizzato da EASYWORD), e sia il cursore posizionato sulla 'P' di 'Pippo'. Eseguiamo il comando, e premiamo 'V': sparisce 'Pippo'. Se premiamo invece 'F' sparisce tutto fino al primo punto. Solo con 'P' si riesce a cancellare l'intero paragrafo.

SH-CTRL-E o **SH-CTRL-2** — Attivando questo comando, descritto appena sopra, il *buffer* non viene pulito. Vediamo con un esempio la differenza con CTRL-E. Consideriamo l'esempio precedente, e utilizziamo il comando SH-CTRL-E: sparisce 'Pippo'. Premiamo 'RETURN,' andiamo alla fine del testo (CTRL-Z) e premiamo CTRL-R (che come vedremo recupera il contenuto del *buffer*): otterremo sul video 'Pippo'. Torniamo ora all'inizio della frase, adesso tronca, e premiamo nuovamente SH-CTRL-E cancellando il vocabolo 'accese'. Portiamoci nuovamente alla fine del testo ed eseguiamo ancora CTRL-R: vedremo comparire 'Pippo accese'. Ripetiamo ora la stessa operazione utilizzando il comando CTRL-E: la prima volta otteniamo, come con SH-CTRL-E, 'Pippo', ma la seconda il *buffer* contiene solo la parola 'accese': il contenuto precedente del *buffer* è andato perso. L'utilità di questo duplice comando di eliminazione può essere chiaro con il seguente esempio: sia la frase 'ma non è sempre vero'; per cambiarla in 'non sempre è vero' bisogna posizionarsi all'inizio, usare CTRL-E (o anche SH-CTRL-E) e cancellare 'non', portare il cursore sul verbo, premere CTRL-E per cancellare il *buffer* ed eliminare 'è'. Si mette infine il cursore sulla 'v' di 'vero' e si preme CTRL-R. Se invece vogliamo cambiare la frase 'non è una bella casa' in 'una casa non è bella', con SH-CTRL-E si può eliminare prima 'una', poi 'casa' e il gioco è fatto, poiché il buffer contiene ora 'una casa'. Infine bisogna ricordare che il buffer ha dimensioni limitate: 255 caratteri che generalmente sono più che sufficienti per l'eliminazione di qualsiasi paragrafo; quando però esso è completamente riempito di caratteri, non è più possibile eliminare nulla: appare la scritta

Buffer pieno

e bisogna o svuotarlo (CTRL-P) o effettuare lo spostamento in due volte.

CTRL-W — Permette di cancellare del testo prima del cursore, mettendolo nel *buffer* che viene prima cancellato (analogamente a CTRL-T) EASYWORD mostra la scritta

Togli (V,F,P)

Per l'eliminazione di un vocabolo, di una frase o di un paragrafo. Per uscire dal comando è sufficiente digitare un tasto diverso dai tre che operano la cancellazione.

SH-CTRL-W — Il comando si comporta esattamente come CTRL-W, se escludiamo il fatto che, prima di operare la cancellazione, il *buffer* non viene svuotato del contenuto precedente. Come per i comandi di eliminazione di testo successivo, anche per CTRL-W e SH-CTRL-W la cancellazione può avvenire solo a condizione che il buffer non sia pieno.

CTRL-R — È il comando che permette di rimediare agli errori di cancellazione e che realizza gli spostamenti di testo. Il suo effetto è quello di 'svuotare' sullo schermo il contenuto del *buffer*, spostando eventualmente tutto il resto del testo dopo il cursore. Da notare che il *buffer* non viene cancellato, quindi il comando può essere utilizzato per inserire quante volte si vuole, fino al riempimento della memoria del testo, cosa segnalata da EASYWORD mediante il messaggio:

Fine memoria

Solo pochi wordprocessor possiedono un comando di tale utilità. Vediamo ora come utilizzarlo per duplicare parte di testo. Sia per esempio la frase 'era una bella casa non molto pulita, ma che vogliamo rendere 'era una bella casa non molto pulita, ma era una bella casa'. Bisogna posizionarsi all'inizio, premere CTRL-E, quattro volte 'V' (per cancellare i primi quattro vocaboli) e poi 'RETURN' (per uscire dal modo eliminaizone). A questo punto rimante solo la frase 'non molto pulita, ma'; premendo CTRL-R subito e poi dopo esserci portati alla fine di essa, otteniamo ciò che volevamo.

CTRL-£ — Questo è il comando che permette di operare la ricerca e la sostituzione automatica di intere parti all'interno del testo. Prima bisogna definire l'insieme di caratteri da cercare, alla richiesta

Cerca:

Poi vanno dati i caratteri da sostituire, alla domanda di EASYWORD

Sostituisci:

Dopo aver dato il 'RETURN' alla seconda domanda, EASYWORD esegue automaticamente il comando, iniziando la ricerca dalla posizione del cursore. Da notare che non viene dato alcun messaggio al termine: se non è stata modificata alcuna parte, il cursore non si sarà mosso, altrimenti si troverà dopo l'ultima sostituzione effettuata. Da notare che è possibile eseguire il comando senza alcuna paura, non andando mai esso in circoli viziosi quali ad esempio si potrebbero ottenere sostituendo 'p' con 'pp'. Infatti la sostituzione è tale che il cursore ogni volta si posiziona alla fine del testo appena immesso, in modo da non considerarlo alla ricerca successiva.

SH-CTRL-H — Con questo e il prossimo comando è possibile posizionare il cursore ad un particolare brano, in maniera automatica. SH-CTRL-H permette di definire lo scritto (può essere però anche solo un carattere) da ricercare.

Cerca:

è il messaggio di EASYWORD.

CTRL-H — Permette la ricerca automatica all'interno del testo, a partire dalla posizione del cursore, della prima occorrenza dei caratteri definiti mediante SH-CTRL-H. In caso positivo, il cursore si posiziona alla prima lettera della serie ricercata, altrimenti esso non si sposta e compare il messaggio

Negativo

Una volta definita con SH-CTRL-H una serie di caratteri, il comando CTRL-H può essere utilizzato quante volte si vuole, a meno di avere nel frattempo usato CTRL-£.

SH-CTRL-J — Con questo comando (da usare congiuntamente al successivo) è possibile definire uno o più caratteri da sostituire alla parola trovata con CTRL-H. Questa va fornita alla richiesta

Sostituisci:

di EASYWORD.

CTRL-J — Effettua la modifica del testo trovato da CTRL-H con quello precedentemente definito da SH-CTRL-J. Questo comando agisce solo congiuntamente a CTRL-H, ed è estremamente utile in casi tipo il seguente: supponiamo di aver scritto la frase 'la tavola purtropo non si può aparecchiare', come potete vedere a volte alcune 'p' vanno raddoppiate ed altre no. Il modo più veloce per la correzione è quello di ricorrere a CTRL-H e CTRL-J. Si comincia col definire 'p' mediante SH-CTRL-H e 'pp' con SH-CTRL-J, quindi si posizio-

na il cursore all'inizio e si esegue CTRL-H: ogni volta che è necessario premiamo CTRL-J, altrimenti continueremo con CTRL-H fino al termine. Facile no?

1.4 I codici di formato di EASYWORD

EASYWORD fornisce la possibilità di controllare completamente il formato di stampa, in modo da poterne fissare i parametri nella maniera che si ritiene migliore. Ad esempio è possibile determinare la lunghezza delle righe di stampa, o il numero di esse che vanno stampate per foglio. Questa caratteristica rende EASYWORD sicuramente uno dei più versatili wordprocessor per il Commodore 16, permettendo il suo utilizzo realmente in ogni ambiente, studio o lavoro che sia.
I comandi di formato non vengono stampati, ma servono unicamente al programma per eseguire la stampa; essi trovano posto nel testo, assieme allo scritto da stampare, ma si distinguono da esso per il fatto di essere mostrati sul video in *reverse*. Per essere ottenuti bisogna utilizzare il comando CTRL-F, e in generale sono costituiti da un solo carattere, cui può seguire un valore numerico.
Possiamo suddividere tali direttive di stampa in due categorie:

— **comandi globali**: sono quelli che vengono eseguiti prima che una linea venga stampata e controllano le variabili di stampa. Queste riassumono le caratteristiche della carta (lunghezza, larghezza) e del tipo di stampa che si vuole ottenere (marginatura, spaziatura, ecc.). Tali comandi non devono essere mischiati al testo, ma possono però stare più d'uno in una sola riga. Le variabili di stampa hanno un proprio valore di *default*, che cioé assumono in caso non venga utilizzato l'opportuno comando di modifica, quindi non è necessario specificarle sempre tutte: i comandi globali servono solo per variarne, quando utile, il valore.

— **comandi locali**: queste direttive permettono di modificare solo la linea che viene stampata al momento (per la sottolineatura o il centraggio), e non modificano le variabili di stampa. Possono trovare posto ovunque all'interno del testo, sia tra lo scritto che tra i comandi di stampa globali.

Nei prossimi due sottoparagrafi sono spiegati tutti i comandi di formato: sintassi ed effetto. In figura 2 è presentato un esempio di testo di EASYWORD (potete provare a digitarlo per prova) che utilizza gran parte delle direttive di formato più importanti, e in figura 3 è mostrato l'output che si ottiene su una stampante a margherita. In questo modo si spera di risolvere tutti i dubbi che possono essere legati a tali particolari comandi che rendono EASYWORD un eccellente wordprocessor.

1.4.1 I codici di formato globali

l: (sta per 'left', sinistra), *deafult* 5; permette di regolare il margine sinistro, cioè il numero di spazi che deve essere lasciato all'inizio di ogni riga di stampa. Il suo valore può variare tra 0 e 255.

r: (sta per 'right', destra), *deafult* 75; consente di fissare il numero di colonna oltre al quale lo scritto non può andare. Il suo valore, tra 0 e 255, deve essere maggiore di quello fissato per il margine sinistro, e la lunghezza delle linee stampate è data dalla differenza di tali valori (quindi per *deafult* essa è 70).

t: (sta per 'top', alto), *deafult* 5; è il numero di linee dal bordo superiore della carta dopo le quali inizia la stampa. Perché questo comando sia veramente utile, è necessario assicurarsi, prima di iniziare la stampa, che la testina della stampante sia posizionata esattamente sulla separazione tra due fogli.

b: (sta per 'bottom', fondo), *deafult* 66; indica l'ultima linea della carta che viene stampata. La differenza con il parametro precedente dà il numero totale di righe di testo per pagina.

p: (sta per 'page', pagina) *deafult* 72; è la variabile che contiene la lunghezza dei fogli su cui avverrà la stampa, espressa in numero di linee. Generalmente la lunghezza normale è 72, anche se la carta standard americana è leggermente più corta (66 righe). Variando tale parametro è opportuno rivedere di conseguenza anche il precedente.

x: *deafult* 80; è la larghezza della pagina, cioè il massimo valore ammesso come fine testo. Generalmente il formato più comune della carta utilizzata in Italia è 72 righe per 80 colonne.

m: disabilita il margine sinistro della successiva linea stampata. In pratica la fa iniziare dalla prima colonna. Questo comando, che non deve essere seguito da alcun valore, deve essere inserito nel testo prima del paragrafo che va identato esternamente e la sua azione si riferisce solo ad esso: i paragrafi successivi saranno stampati, come gli altri, considerando il valore del margine sinistro.

h: (sta per 'header', testata); seguito da una qualsiasi scritta di massimo 254 caratteri, questo comando realizza la stampa della scritta sulla prima riga di ogni foglio (quindi sopra il testo), in modo da potersi riferire in questo modo, per esempio, all'argomento trattato dal documento. La scritta può anche essere centrata (ved. comando di formato 'c'). Bisogna fare attenzione che la variabile 't' (margine superiore del testo) permetta l'utilizzo della testata.

f: (sta per 'footer', fondo); permette, analogamente a 'h', la stampa di una qualsiasi scritta (di al massimo 254 caratteri da digitare subito dopo il comando) a fondo pagina sotto il testo, all'ultima riga della pagina. Anche in questo caso bisogna verificare che il valore di 'b' (margine inferiore del testo) permetta le scritte a fondo pagina. Sia la testata che la scritta a pie' di pagina compariranno dal successivo foglio stampato fino al termine del testo.

s: *deafult* 1; è la spaziatura, cioè il numero di linee da incrementare dopo ogni linea: un valore 0 non fa mai avanzare la carta, mentre un valore 2 lascia una riga bianca tra una linea e la successiva.

n: (sta per 'next', seguente); salto pagina forzato. Non è richiesto alcun numero successivo. Quando EASYWORD incontra questo comando in fase di stampa, la riga successiva viene stampata su nuova pagina.

@: EASYWORD numera automaticamente le pagine stampate a partire da uno: per iniziare da un numero qualsiasi si utilizza questo comando, cui deve seguire il numero della pagina da stampare. Per esempio '@3' farà in modo che la prima pagina stampata abbia il numero tre.

?: disabilita la stampa fino a che non deve essere stampata la pagina corrispondente al numero che segue il comando: per esempio '**?**3' non stampa le prime due pagine.

i: (sta per 'informazioni'); permette di inserire all'interno del testo un commento, fino a 255 caratteri da digitare dopo il comando, che non verrà stampato.

w: (sta per 'wait', attendi); questo comando, che deve essere posto all'inizio del testo e che una volta abilitato non può più essere disinserito, permette la stampa su fogli singoli, qualora la stampante lo permetta (per esempio la MPS 802). Al termine della stampa di ogni singolo foglio, EASYWORD mostra nella finestra di comando il messaggio:

Inserisci foglio e premi RETURN

e rimane in attesa della pressione del tasto. La sua utilità è evidente qualora si volesse utilizzare una carta di qualità o intestata. Questo comando viene ignorato in caso la stampa avvenga su schermo o su disco.

j: è il *linefeed* automatico, e dev'essere posto all'inizio del testo. La sua utilità si manifesta con l'uso di stampanti diverse da quelle Commodore, nelle quali l'avanzamento della carta al termine di ogni riga stampata (il *linefeed*) non è dato automaticamente. È da utilizzarsi solo nel caso la propria stampante stampi il testo su un'unica riga senza andare mai a capo.

a: (sta per 'ASCII', American Standard Code for Information Interchange); permette di utilizzare stampanti non Commodore e va posto all'inizio del testo. La Commodore ha infatti realizzato un protocollo di comunicazione computer-stampante che è leggermente diverso da quello standard mondiale ASCII. Per ottenere un output assolutamente compatibile con lo standard ASCII è necessario questo comando, che permette quindi l'utilizzo di qualsiasi stampante. L'unica attenzione da fare in questo caso è quella di non stampare caratteri grafici Commodore, che non sono assolutamente standard ASCII.

g: (sta per 'go to', vai a); è un comando fondamentale poiché permette il concatenamento di *file* di testo in fase di stampa. Il comando va posto come ultima riga del testo, ed ha il seguente formato:

gp:nomefile

dove p ('periferica') deve essere 'd' o 'n' a secodna che si utilizzi il disco o il nastro, mentre nomefile è il nome del *file* su cui è registrato il proseguimento dello scritto (va messo senza le 'virgolette'). EASYWORD al termine della stampa, carica dalla periferica indicata il *file* segnalato, e quindi riprende a stampare come se nulla fosse: tutti i comandi di formato precedenti (quali numero pagina, marginatura, ecc.) continuano a valere. Se anche alla fine dello scritto caricato c'è un'istruzione 'g', questa viene eseguita, ed in questo modo è possibile stampare senza interruzioni un numero illimitato di *file*. Da notare che questa direttiva ha come effetto il caricamento del *file* richiamato in memoria, con conseguente perdita di quello precedente, quindi bisogna fare attenzione anche quando si vuole vedere la stampa su video: è opportuno sempre registrare prima su nastro (o disco) i propri scritti.

```
Prova utilizzo codici di formato←
E←
Questa riga non va stampata←
←
PAGINA ←
:80←
1234567890123456789012345678901234567890
1234567890123456789012345678901234567890←
75←
Parto dalla sesta colonna e termino
solo alla settantacinquesima, anche se
non ci arrivo perche' vado a capo
prima!←
3:46←
Io invece parto dalla quarta e finisco
molto prima: appena alla
quarantaseiesima←
5:75:←
Io comincio subito!←
Questo e' un commento←
2←
Spaziatura due!←
Spaziatura due! (sottolineata)←
Spaziatura due!←
1←
Spaziatura uno!←
Spaziatura uno!←
Spaziatura uno!←
=75←
Prova stampa freccia a sinistra:←
```

Figura 2
Testo di prova di alcuni codici di formato

```
                         Prova utilizzo codici di formato


12345678901234567890123456789012345678901234567890123456789012345678901234567890
     Parto dalla sesta colonna e termino solo alla settantacinquesima,
     anche se non ci arrivo perche' vado a capo prima!
   Io invece parto dalla quarta e finisco
   molto prima: appena alla quaranteseiesima
Io comincio subito!
     Spaziatura due!

     Spaziatura due! (sottolineata)

     Spaziatura due!

     Spaziatura uno!
     Spaziatura uno!
     Spaziatura uno!
     Prova stampa freccia a sinistra:←
```

Figura 3
Output del testo di figura 2
ottenuto con 'Easyword'

1.4.2 I codici di formato locali

u: (sta per 'underline', sottolinea); posto sia all'inizio che alla fine di una parola o di una frase, in fase di stampa ne opera la sottolineatura. Questo comando non funziona con le stampanti Commodore, ma solo con tutte quelle che permettono lo spostamento del carrello di un carattere indietro (ottenuto mediante l'invio di CHR$ (8)) e la sottolineatura (CHR$(95)). Infatti l'effetto è realizzato facendo seguire ad ogni carattere della parole (o frase) i due caratteri-stringa 8 e 95.

c: (sta per 'centratura'); permette di centrare in mezzo al foglio la scritta di testo fino al primo 'RETURN'. Essa avviene calcolando il valore dei parametri di stampa 'l' e 'r', in modo che la centratura sia sempe rispetto al testo scritto e non al foglio.

#: ogni volta che EASYWORD incontra questo comando, stampa al suo posto il numero della pagina attualmente in stampa: è molto utile per la numerazione automatica mediante il *footer*. Il valore stampato è influenzato da un eventuale comando '@'.

1,2,... ,9: caratteri definibili. Molte stampanti usano codici particolari per effettuare speciali funzioni quali sottolineatura o caratteri elongati: essi sono riassunti nel libretto d'utilizzo che accompagna ogni macchina. Generalmente questi codici di controllo hanno un valore ASCII minore di 32; tali valori non sono direttamente ottenibili tramite tastiera, e per questo motivo EASYWORD permette di definirne nove diversi, eventualmente ridefinibili nel corso della stampa. Ad esempio non è possibile stampare direttamene il simbolo '←' (il cui codice ASCII è 95), poiché all'interno di EASYWORD esso viene utilizzato come simbolo del 'RETURN'. In questo caso è sufficiente definire il tasto di formato 1 mediante:

1= 95

per ottenere, ogni volta che compare il simbolo '1', la freccia a sinistra.

1.5 Come funziona EASYWORD

Lo scopo principale di questo libro è si quello di fornire al lettore dei potenti strumenti di lavoro, ma è anche quello di aumentare la sua conoscenza riguardo il computer ed il suo utilizzo. Per questo motivo tutti i programmi sono esaurientemente descritti e commentati in modo da permettere a chiunque, dotato di un po' di buona volontà, la totale comprensione del loro funzionamento. Bisogna ricordare infatti che il miglior modo per perfezionare la propria capacità programmatoria è proprio quello di studiare i casi particolari (cioè i programmi degli altri) cercando di comprendere i motivi che hanno portato alle singole scelte.
EASYWORD è interamente scritto in assembler, e vediamo cosa significhi ciò. Un computer può essere visto come una persona con la quale sia possibile comunicare soltanto tramite una tastiera: le sue possibilità, come ben sappiamo, sono vastissime e ampio è il campo delle sue applicazioni. Ci sono però due grossi problemi: il primo è che questa persona è assolutamente stupida, nel senso che si limita ad eseguire solo le istruzioni che gli vengono date senza metterci niente di proprio. Ciò può essere comodo in molti casi, ma in altri è addirittura disarmante: se chiediamo ad un individuo umano di calcolare una difficile operazione, è implicito il fatto che poi volgiamo sapere il risultato, ed egli infatti ce lo darà al termine del calcolo. Non così il computer: se non gli viene esplicitamente chiesto il risultato, esso se lo tiene per sé!
Il secondo problema di comunicazione con la persona-calcolatore è costituito dal linguaggio che viene da essa compreso: a ben guardare esso è costituito da sole 56 parole (per il Commodore 16), al profano assolutamente incomprensibili. L'insieme di queste parole costituisce quello che viene chiamato il 'codice macchina': è il linguaggio numerico compreso dal microprocessore che realizza il funzionamento del computer; il Commodore 16 utilizza l'integrato MOS 8501 come unità di microprocessione. Ogni singolo comando permette però una azione

di *basso livello*: così vengono chiamate tutte quelle istruzioni che agiscono sulle singole celle di memoria; quindi per realizzare un'istruzione di *alto livello* (quale per esempio una moltiplicazione tra grossi numeri o la stampa del risultato) sono necessarie più istruzioni in linguaggio macchina, rendendo la programmazione, cioè la stesura dei programmi, un compito assai arduo: basti pensare che per realizzare una semplice moltiplicazione tra numeri bassi sono necessarie una ventina di istruzioni del 8501!
Per questo motivo sono nati i linguaggi di programmazione, cioè degli insiemi di parole chiave che sono comprese da un *interprete* che le traduce in codice macchina: FORTRAN, COBOL, BASIC e infine PASCAL sono degli esempi di liguaggi di alto livello. In particolare il BASIC, per la sua facilità di apprendimento e di utilizzo, è il linguaggio divenuto dominante nei personal e home computer. Tutti questi linguaggi, però, necessitano come abbiamo visto di un altro programma, il cosiddetto interprete, per poter essere compresi dal microprocessore, e ciò ne rallenta notevolmente la velocità: ad ogni istruzione del programma, il processore deve attendere che il traduttore compia la sua funzione.
Ciò spiega l'importanza dei linguaggi di basso livello: anche se, a parità di computazione, i programmi di questo tipo appaiono molto più lunghi dei corrispondenti di alto livello, la loro velocità di esecuzione è di molto minore. Il linguaggio a basso livello più immediato (se vogliamo quello a livello 'minore') è l'*assembler*, cioè la traduzione dei singoli codici del linguaggio macchina in opportuni mnemonici, in modo da facilitarne l'utilizzo. Tale linguaggio dipende quindi direttamente dal particolare processore, ed esiste un assembler per ogni tipo di unità di calcolo: le diversità tra assembler diversi possono essere anche notevoli.
Come detto, EASYWORD è scritto interamente in assembler, e quindi sono le istruzioni di questo linguaggio che sono in seguito spiegate. Non dovete però preoccuparvi dell'iniziale difficoltà di comprensione del linguaggio: dopo un po' di pratica scoprirete la potenza e la velocità dell'assembler e avrete a disposizione un'eccezionale strumento di programmazione per ogni vostra futura applicazione. Naturalmente non tutti sono a conoscenza del funzionamento dell'assembler, e per questo motivo prima del commento al programma abbiamo ritenuto utile descriverne in breve le caratteristiche fondamentali.

1.5.1 L'assembler del 8501

In questo paragrafo vediamo in breve la struttura dell'assembler del 8501, il microprocessore del vostro Commodore 16. Non approfondiamo eccessivamente il discorso, ma rimandiamo ai testi specifici per tutti quegli aspetti, alcuni in verità fondamentali, che in questa sede sono stati considerati superflui agli scopi prefissati.
Due sono i concetti che devono essere ben chiari per comprendere il funzionamento di un calcolatore: gli indirizzi di memoria e il *file*. L'esperienza già dovrebbe aver chiarito il significato del *file*, o archivio: è l'insieme dei dati che permettono

di realizzare uno specifico compito. Ai fini della comprensione dell'assembler è però fondamentale il primo concetto: le locazioni di memoria. Il Commodore 16 ha 6553 celle di memoria, in ognuna della quali può stare un numero tra 0 e 255, o meglio otto bit che possono valere o 0 o 1. Tutte le locazioni considerate globalmente caratterizzano lo stato della macchina: ad esempio quelle da 3072 a 4095 contengono la mappa video, cioè i caratteri che compaiono di volta in volta sullo schermo. Il microprocessore 8501 può leggere, interpretare e modificare il contenuto di gran parte di esse. Un programma in linguaggio macchina è costituito da un insieme generalemtne contiguo di locazioni di memoria: lo 8501 le considera una dopo l'altra come fossero un normale programma BASIC, solo che ora non esiste più una numerazione: 'GOTO numero linea', per intenderci, diventa 'GOTO locazione di memoria'.
Per realizzare il proprio compito, il 8501 ha una sua particolare struttura interna, costituita da tre registri, che non sono altro che celle di memoria: l'accumulatore (o registro A), che permette i calcoli,e i registri X e Y. Il microprocessore utilizza tali registri per poter leggere e scrivere in memoria. Gran parte delle 56 istruzioni dell'assembler sono costituite da un mnemonico più un operando: esso è costituito da una o due locazioni di memoria. Ad esempio per poter caricare nell'accumulatore il numero 4 sono necessarie 2 locazioni di memoria: la prima deve contenere il numero corrispondente all'azione di caricamento (160), ed il secondo il valore (4).
Vediamo ora in breve tutto il set di istruzioni dell'assembler del 8501, fornendo un breve commento di ognuna:

ADC Somma il bit di carry con il contenuto dell'accumulatore e con il valore numerico seguente (può anche indicare la cella di memoria nella quale si trova il numero). Il risultato rimane nell'accumulatore.
AND Esegue l'operazione di AND logica tra il contenuto dell'accumulatore e un dato specifico. Il risultato rimane nell'accumulatore.
ASL Muove gli 8 bit dell'accumulatore o della locazione specificata di un posto verso sinistra; da destra entra uno 0 mentre il bit 7 va nel bit di carry (C). Il risultato viene mantenuto. Il bit C segnala un riporto numerico durante una somma o una sottrazione oppure contiene un bit di una particolare cella.
BCC Va all'indirizzo specificato nel caso C sia 0.
BCS Va all'indirizzo specificato nel caso C sia 1.
BEQ Va all'inirizzo specificato nel caso il bit di zero (Z) sia 1. Il bit Z, se uguale a 1, indica che l'ultimo trasferimento o calcolo è stato uno zero.
BIT Viene eseguita l'operazione di AND logica tra il contenuto dell'accumulatore e il dato specificato. Il risultato non viene conservato, ma viene modificato, tra gli altri, il bit Z.
BMI Va all'inirizzo specificato nel caso il bit negativo (N) sia 1. Tale bit è 1 ogni volta che l'ultimo dato trattato è negativo nella rappresentazione in complemento a due.
BNE Va all'indirizzo specificato nel caso Z sia 0.

BPL Va all'indirizzo specificato nel caso N sia 0.

BRK Interrompe l'esecuzione di un programma in assembler.

BVC Va all'indirizzo specificato nel caso il bit di overflow (V) sia 0. Il bit V, se uguale a 1, indica che l'ultima operazione può non essere corretta (numeri in rappresentazione in complemento a due).

BVS Va all'indirizzo specificato nel caso V sia 0.

CLC Azzera il bit C.

CLD Azzera il bit D, che permette di passare dal modo BCD (1) al modo binario (0).

CLI Azzera il bit di interrupt (I). Il bit I, se 0, permette la possibilità di sospendere l'esecuzione di un programma per eseguire un'opportuna routine in codice macchina.

CLV Azzera il bit V.

CMP Il dato specificato viene sottratto dal contenuto dell'accumulatore. Il risultato non viene conservato, ma l'operazione influenza i bit N, Z e C.

CPX Il dato specificato viene sottratto dal contenuto del registro X. Il risultato non viene conservato, ma l'operazione influenza i bit N, Z e C.

CPY Il dato specificato viene sottratto dal contenuto del registro Y. Il risultato non viene conservato, ma l'operazione influenza i bit N, Z e C.

DEC Il contenuto della cella specificata viene decrementato di uno, e il risultato viene memorizzato al posto del precedente.

DEX Il contenuto del registro X viene decrementato di uno, e il risultato viene memorizzato al posto del precedente.

DEY Il contenuto del registro Y viene decrementato di uno, e il risultato viene memorizzato al posto del precedente.

EOR Esegue l'operazione di OR esclusivo tra il contenuto dell'accumulatore e un dato specifico. Il risultato rimane nell'accumulatore.

INC Il contenuto della cella specificata viene incrementato di uno, e il risultato viene memorizzato al posto del precedente.

INX Il contenuto del registro X viene incrementato di uno, e il risultato viene memorizzato al posto del precedente.

INY Il contenuto del registro Y viene incrementato di uno, e il risultato viene memorizzato al posto del precedente.

JMP Va all'indirizzo specificato.

JSR Esegue la subroutine che parte dall'indirizzo specificato. È il 'GOSUB' dell'assembler.

LDA Carica nell'accumulatore il dato specificato.

LDX Carica nel registro X il dato specificato.

LDY Carica nel registro Y il dato specificato.

LSR Muove gli 8 bit dell'accumulatore o della locazione di memoria specificata di un posto verso destra; da sinistra entra uno 0 mentre il bit 0 va nel bit di carry. Il risultato viene mantenuto.

NOP Non opera: può servire per riempire dei 'buchi' all'interno di un programma.

ORA Esegue l'operazione di OR logico tra il contenuto dell'accumulatore e il dato specificato. Il risultato rimane nell'accumulatore.
PHA Il contenuto dell'accumulatore viene spinto nello stack, il cui puntatore viene incrementato. A rimane invariato.
PHP Il contenuto del registro di stato viene spinto nello stack, il cui puntatore viene incrementato. A rimane invariato.
PLA Il contenuto dell'estremità superiore dello stack viene depositato nell'accumulatore. Il puntatore di stack viene aggiornato.
PLP Il contenuto dell'estremità superiore dello stack viene depositato nel registro di stato. Il puntatore di stack viene aggiornato.
ROL Gli 8 bit dell'accumulatore o della locazione di memoria specificata sono spostati di una posizione a sinistra. Il contenuto del carry va nel bit 0, mentre il bit 7 viene trasferito in C.
ROR Gli 8 bit dell'accumulatore o della locazione di memoria specificata sono spostati di una posizione a destra. Il contenuto del carry va nel bit 7, mentre il bit 0 viene trasferito in C.
RTI È l'istruzione di ritorno da interrupt al programma che il microprocessore stava eseguendo prima di esso.
RTS Ritorno da subroutine al programma chiamante.
SBC Sottrae dall'accumulatore il bit C e il contenuto della locazione di memoria specificata. Il risultato rimane nell'accumulatore.
SEC Il bit C viene posto a 1.
SED Il bit D viene posto a 1.
SEI Il bit I viene posto a 1.
STA Il contenuto di A viene ricopiato nella locazione di memoria specificata. A non viene modificato.
STX Il contenuto di X viene ricopiato nella locazione di memoria specificata. X non viene modificato.
STY Il contenuto di Y viene ricopiato nella locazione di memoria specificata. Y non viene modificato.
TAX Trasferisce il contenuto dell'accumulatore nel registro X. A non viene alterato.
TAY Trasferisce il contenuto dell'accumulatore nel registro Y. A non viene alterato.
TSX Trasferisce il contenuto del puntatore di stack nel registro X. Il puntatore non viene alterato.
TXA Trasferisce il contenuto del registro X nell'accumulatore. X non viene alterato.
TXS Trasferisce il contenuto del registro X nel puntatore di stack. X non viene alterato.
TYA Trasferisce il contenuto del registro Y nell'accumulatore. Y non viene alterato.

Siamo ora in grado di poter almeno comprendere a grandi linee il funzionamento di un programma assembler. EASYWORD viene 'sezionato' in ogni sua parte nel prossimo paragrafo. Per descrivere ogni particolare è stato utilizzato un particolare disassemblatore (cioè un programma che permette di 'listare' le istruzioni in linguaggio macchina), il cui funzionamento è assai simile e quello ottenibile

mediante il *monitor* sul vostro Commodore 16: ogni linea è del tipo

```
10A8 A9 08      LDA #$08
```

ove il primo numero (10A8) indica il numero della locazione di memoria dell'istruzione considerata (LDA), il cui valore è A9. Il terzo numero (08) indica l'operando associato all'istruzione: esso può non esserci come può anche essere costituito da due numeri. Tutti i numeri sono rappresentati in base 16, essendo in questo modo possibile indicare i numeri tra 0 e 255 con solo due cifre: da 00 a FF, utilizzando le lettere da A a F per coprire i numeri oltre il 9.
Per ottenere sul vostro video le istruzioni disassemblate di EASYWORD, dovete caricare il programma dalla cassetta allegata mediante

LOAD "EASYWORD"   +'RETURN'

o anche più semplicemente

LOAD+ 'RETURN'

se la cassetta ha il nastro riavvolto. Quindi dovete digitare

MONITOR   +'RETURN'

per entrare in 'modo monitor' e quindi premere ad esempio

D 100D 1020

per ottenere i mnemonici di tutte le istruzioni comprese tra le locazioni di memoria 100D e 1020.

1.5.2 Commento al programma

Dopo il caricamento in memoria del programma, l'unica istruzione BASIC che appare è

```
10 SYS4109
```

che passa il controllo al linguaggio macchina: d'ora in poi è il programma assembler che qui viene presentato che si occupa di tutto. Esso utilizza alcune routine del sistema operativo del Commodore 16, tutte di input/output: esistendo infatti già tali routine era inutile scriverne altre equivalenti. È questa una prima caratteristica dell'assembler: è utile adoperare tutte le routine che sono già realizzate per il funzionamento del computer, che sono veramente tante. In

particolare EASYWORD utilizza le seguenti routine (tra parentesi è indicata la locaizone di partenza, in esadecimale: d'ora in poi scordatevi dei numeri in base 10!):

CHROUT	($FFD2) Spedisce un carattere alla periferica di output selezionata (video, stampante, disco o registratore).
STOP	($FFE1) Routine che guarda se è stato premuto il tasto RUN/STOP.
SETLFS	($FFBA) Fissa il numero logico del file (tra 0 e 255), il numero della periferica (0=tastiera, 1=nastro, 3=video, 4=stampante, 8=disco) e il numero secondario di attivazione della periferica.
SETNAM	($FFBD) Fissa il nome del *file* per le operazioni di LOAD, VERIFY e SAVE.
CLALL	($FFE7) Chiude un *file* logico al termine di un'operazione di input/output.
OPEN	($FFC0) Apre un *file* logico. Deve essere eseguita prima di ogni routine di input/output.
CHRIN	($FFCF) Ottiene un carattere dal canale selezionato (può essere la tastiera, l'unità a dischi o il registratore).
CHKIN	($FFC6) Seleziona come canale di input il canale logico definito da OPEN.
CHKOUT	($FFC9) Seleziona come canale di output il canale logico definito da OPEN.
GETIN	($FFE4) Ottiene un carattere dalla periferica di input, nel caso esso esista.
CLRCHN	($FFCC) Inizializza i canali di input/output aperti e le periferiche connesse. Di solito si utilizza prima di un'operazione di 1/0.
CLOSE	($FFC3) Chiude un *file* logico.
LOAD	($FFD5) Carica in memoria dalla periferica selezionata.
SAVE	($FFD8) Salva sulla periferica selezionata parte di memoria.
IOINIT	($FF84) Inizializza tutte le periferiche connesse e tutti canali di input/output.

E veniamo finalmente al commento del programma: le prime istruzioni che vengono eseguite sono

```
100D 20 88 11 JSR $1188
1010 A9 CA    LDA #$CA
1012 CD 6E 2B CMP $2B6E
1015 8D 6E 2B STA $2B6E
```

```
1018 F0 03    BEQ $101D
101A 20 37 11 JSR $1137
101D 20 C5 11 JSR $11C5
1020 4C 69 12 JMP $1269
```

Il programma assembler è strutturato in modo da essere costituito soltanto da subroutine, chiamate da un *main*, un programma principale, che vedremo in seguito. Ogni routine ha un nome, naturalmente di fantasia, che cerca di spiegare la sua azione. Questa è la routine **INIZIO** che viene chiamata dopo il 'RUN' al programma, e attiva la subroutine di inizializzazione INIT ($1188 è il suo indirizzo di partenza). Quindi scrive in una locazione di memoria libera le iniziali dell'autore: si suppone che la prima volta che EASYWORD viene eseguito esse non siano presenti; in questo modo si riesce ad evitare la cancellazione del testo (CANCELLA, $1137) nel caso si sia usciti dal programma (ciò può essere ottenuto premendo il tasto di reset unitamente a RUN/STOP). il JMP finale è al MAIN ($1269) del programma.

```
1023 A5 DB    LDA $DB
1025 8D 43 10 STA $1043
1028 A5 DC    LDA $DC
102A 8D 44 10 STA $1044
102D A5 DD    LDA $DD
102F 8D 46 10 STA $1046
1032 A5 DE    LDA $DE
1034 8D 47 10 STA $1047
1037 A6 E1    LDX $E1
1039 F0 20    BEQ $105B
103B A9 00    LDA #$00
103D 8D 15 28 STA $2815
1040 A0 00    LDY #$00
1042 B9 00 00 LDA $0000,Y
1045 99 00 00 STA $0000,Y
1048 C8       INY
1049 CC 15 28 CPY $2815
104C D0 F4    BNE $1042
104E EE 44 10 INC $1044
1051 EE 47 10 INC $1047
1054 E0 00    CPX #$00
1056 F0 07    BEQ $105F
1058 CA       DEX
```

```
1059 D0 E0     BNE $103B
105B A5 E0     LDA $E0
105D D0 DE     BNE $103D
105F 60        RTS
```

Questa routine, **MUOVISU** realizza un rapido spostamento di memoria, ricopiando parte di memoria sopra memoria successiva. I parametri sono fissati dalla stessa routine, che scrive nelle locazione $1043, $1044, $1046 e $1047 gli indirizzi su cui leggere e scrivere. È questo un esempio di autoscrittura, molto utilizzato nella programmazione in assembler: è il programma stesso che decide cosa farà in seguito. $DB, $DD e $E1 contengono l'indirizzo di partenza dell'area di memoria da spostare, l'indirizzo iniziale da cui bisogna scrivere e la lunghezza, in byte, della memoria da trasferire.

```
1060 A5 E1     LDA $E1
1062 AA        TAX
1063 05 E0     ORA $E0
1065 D0 01     BNE $1068
1067 60        RTS
1068 18        CLC
1069 8A        TXA
106A 65 DC     ADC $DC
106C 8D 8B 10  STA $108B
106F A5 DB     LDA $DB
1071 8D 8A 10  STA $108A
1074 18        CLC
1075 8A        TXA
1076 65 DE     ADC $DE
1078 8D 8E 10  STA $108E
107B A5 DD     LDA $DD
107D 8D 8D 10  STA $108D
1080 E8        INX
1081 A4 E0     LDY $E0
1083 D0 04     BNE $1089
1085 F0 0D     BEQ $1094
1087 A0 FF     LDY #$FF
1089 B9 00 00  LDA $0000,Y
108C 99 00 00  STA $0000,Y
108F 88        DEY
```

```
1090 C0 FF    CPY #$FF
1092 D0 F5    BNE $1089
1094 CE 8B 10 DEC $108B
1097 CE 8E 10 DEC $108E
109A CA       DEX
109B D0 EA    BNE $1087
109D 60       RTS
```

MUOVIGIU, opera in modo analogo a MUOVISU, solo che ora la memoria viene spostata verso il basso. Questa routine è quindi utilizzata per inserire del testo in memoria, mentre la precedente serviva per cancellare. Se i due blocchi di memoria (quello da spostare e quello che viene scritto) non si sovrappongono, entrambe le routine possono essere equivalentemente utilizzate.

```
109E A9 28    LDA #$28
10A0 85 D6    STA $D6
10A2 85 D8    STA $D8
10A4 A9 0C    LDA #$0C
10A6 85 D7    STA $D7
10A8 A9 08    LDA #$08
10AA 85 D9    STA $D9
10AC AD 11 28 LDA $2811
10AF 85 3F    STA $3F
10B1 AD 12 28 LDA $2812
10B4 85 40    STA $40
10B6 A2 01    LDX #$01
10B8 AD 14 28 LDA $2814
10BB 85 DA    STA $DA
10BD AD 1D 15 LDA $151D
10C0 20 F0 11 JSR $11F0
10C3 A0 00    LDY #$00
10C5 AD 2C 15 LDA $152C
10C8 91 D8    STA ($D8),Y
10CA B1 3F    LDA ($3F),Y
10CC 99 1D 28 STA $281D,Y
10CF C8       INY
10D0 29 7F    AND #$7F
10D2 C9 1F    CMP #$1F
10D4 F0 13    BEQ $10E9
```

```
10D6 C0 28    CPY #$28
10D8 D0 EB    BNE $10C5
10DA 88       DEY
10DB B1 3F    LDA ($3F),Y
10DD 29 7F    AND #$7F
10DF C9 20    CMP #$20
10E1 F0 05    BEQ $10E8
10E3 88       DEY
10E4 D0 F5    BNE $10DB
10E6 A0 27    LDY #$27
10E8 C8       INY
10E9 84 41    STY $41
10EB 88       DEY
10EC B9 1D 28 LDA $281D,Y
10EF 91 D6    STA ($D6),Y
10F1 88       DEY
10F2 10 F8    BPL $10EC
10F4 A4 41    LDY $41
10F6 18       CLC
10F7 98       TYA
10F8 65 3F    ADC $3F
10FA 85 3F    STA $3F
10FC A5 40    LDA $40
10FE 69 00    ADC #$00
1100 85 40    STA $40
1102 E0 01    CPX #$01
1104 D0 03    BNE $1109
1106 8C 10 28 STY $2810
1109 C0 28    CPY #$28
110B F0 08    BEQ $1115
110D A9 20    LDA #$20
110F 91 D6    STA ($D6),Y
1111 C8       INY
1112 4C 09 11 JMP $1109
1115 18       CLC
1116 A5 D6    LDA $D6
1118 69 28    ADC #$28
111A 85 D6    STA $D6
111C 85 D8    STA $D8
```

```
111E 90 04      BCC $1124
1120 E6 D7      INC $D7
1122 E6 D9      INC $D9
1124 E8         INX
1125 E0 19      CPX #$19
1127 F0 03      BEQ $112C
1129 4C C3 10   JMP $10C3
112C A5 3F      LDA $3F
112E 8D 1B 28   STA $281B
1131 A5 40      LDA $40
1133 8D 1C 28   STA $281C
1136 60         RTS
```

REGOLA é la veloce routine chiamata ogni volta da EASYWORD durante il lampeggio del cursore, ed è quella che realizza alcune funzioni importanti: copia ad esempio sullo schermo l'area di memoria testo indicata da $2811, operando in modo da tenere conto dei margini per il *word-wrapping*, la tecnica che permette di non spezzare una parola a fine linea. Ad ogni fine paragrafo mette il simbolo del 'RETURN' (la freccia a sinistra) e aggiorna il colore dello schermo e del bordo grazie alla routine chiamata in $10C0:

```
11F0 8D 15 FF   STA $FF15
11F3 4C 9D 19   JMP $199D
```

```
199D 8D 19 FF   STA $FF19
19A0 8D 1D 15   STA $151D
19A3 60         RTS
```

Infine REGOLA memorizza l'ultima posizione del cursore in modo che una succesiva routine, CONTROLLA, eviti (con uno *scroll*) che il cursore lampeggiante sparisca dallo schermo.

```
1137 AD 08 28   LDA $2808
113A 85 3F      STA $3F
113C 8D 11 28   STA $2811
113F 8D 17 28   STA $2817
1142 85 39      STA $39
1144 AD 09 28   LDA $2809
```

```
1147 85 40       STA $40
1149 8D 12 28    STA $2812
114C 8D 18 28    STA $2818
114F 85 3A       STA $3A
1151 38          SEC
1152 AD 0B 28    LDA $280B
1155 ED 09 28    SBC $2809
1158 AA          TAX
1159 A9 20       LDA #$20
115B A0 FF       LDY #$FF
115D C6 40       DEC $40
115F 91 3F       STA ($3F),Y
1161 C8          INY
1162 E6 40       INC $40
1164 91 3F       STA ($3F),Y
1166 C8          INY
1167 D0 FB       BNE $1164
1169 E6 40       INC $40
116B CA          DEX
116C D0 F6       BNE $1164
116E 91 3F       STA ($3F),Y
1170 60          RTS
```

È la routine **CANCELLA**, che permette la cancellazione di tutto il testo presente in memoria, mettendo degli spazi (#$20 in esadecimale) nella memoria testo.

```
1171 85 41       STA $41
1173 84 42       STY $42
1175 A0 00       LDY #$00
1177 B1 41       LDA ($41),Y
1179 F0 06       BEQ $1181
117B 20 D2 FF    JSR $FFD2
117E C8          INY
117F D0 F6       BNE $1177
1181 60          RTS
1182 20 E4 FF    JSR $FFE4
1185 F0 FB       BEQ $1182
1187 60          RTS
```

La routine **FINESTRA** stampa sulla prima linea dello schermo, la finestra di comando, il messaggio che EASYWORD deve di volta in volta mostrare. Tutti i messaggi sono posizionati al termine del programma (provate a digitare, in 'modo monitor'

M 2612

e vedrete tutti i messaggi del programma). L'indirizzo basso del messaggio è contenuto nell'accumulatore, quello alto nel registro Y: FINESTRA stampa fino a che trova uno 0 nel messaggio. In $1182 c'è una subroutine che rimane in attesa della pressione di un tasto prima di proseguire.

```
1188 A9 00    LDA #$00
118A 8D 14 28 STA $2814
118D 8D 08 28 STA $2808
1190 8D 0A 28 STA $280A
1193 8D 0C 28 STA $280C
1196 8D 0E 28 STA $280E
1199 8D B0 28 STA $28B0
119C 8D CF 28 STA $28CF
119F A9 2C    LDA #$2C
11A1 18       CLC
11A2 69 01    ADC #$01
11A4 8D 09 28 STA $2809
11A7 38       SEC
11A8 A5 38    LDA $38
11AA E9 01    SBC #$01
11AC 8D 0F 28 STA $280F
11AF 38       SEC
11B0 E9 02    SBC #$02
11B2 8D 0D 28 STA $280D
11B5 38       SEC
11B6 E9 01    SBC #$01
11B8 8D 0B 28 STA $280B
11BB A9 FF    LDA #$FF
11BD 8D AE 28 STA $28AE
11C0 A9 93    LDA #$93
11C2 4C D2 FF JMP $FFD2
```

Ed ecco la subroutine **INIT**, che viene chiamata solo all'inizio del programma: il suo compito è quello di inizializzare la memoria, fissando lo spazio per il testo e per il *buffer*, e alcuni altri *flag* (cioè delle variabili di stato). Al termine le istruzioni $11C0-$11C2 operano una pulizia dello schermo: notate come il termine della subroutine non sia un RTS ma un JMP: ciò perchè il salto è ad un'altra routine (del sistema operativo) che a sua volta termina con un RTS. Questo modo di programmare limita certamente la leggibilità del listato ma permette una compattazione maggiore aumentando la velocità di esecuzione.

```
11C5 20 E2 15 JSR $15E2
11C8 A9 80    LDA #$80
11CA 85 9A    STA $9A
11CC 20 5D 19 JSR $195D
11CF AD 08 28 LDA $2808
11D2 85 39    STA $39
11D4 AD 09 28 LDA $2809
11D7 85 3A    STA $3A
11D9 20 F6 11 JSR $11F6
11DC A9 F6    LDA #$F6
11DE A0 27    LDY #$27
11E0 20 71 11 JSR $1171
11E3 EE 13 28 INC $2813
11E6 4C B1 13 JMP $13B1
```

È questa INIT2, una specie di continuazione della precedente routine, con la differenza che viene chiamata non solo all'inizio. Dopo aver pulito il *buffer* ($15E2), modifica la routine di *interrupt*

```
195D 78       SEI
195E A9 88    LDA #$88
1960 8D 14 03 STA $0314
1963 A9 19    LDA #$19
1965 8D 15 03 STA $0315
1968 58       CLI
1969 A9 FF    LDA #$FF
196B 85 D0    STA $D0
196D 85 D3    STA $D3
196F A9 0B    LDA #$0B
1971 85 D1    STA $D1
```

```
1973 A9 07       LDA #$07
1975 85 D4       STA $D4
1977 AD 2C 15    LDA $152C
197A 8D 3B 05    STA $053B
197D AD 1D 15    LDA $151D
1980 8D 19 FF    STA $FF19
1983 4C 06 12    JMP $1206
```

```
1206 A9 00       LDA #$00
1208 A8          TAY
1209 99 67 45    STA $4567,Y
120C C8          INY
120D C0 32       CPY #$32
120F D0 F8       BNE $1209
1211 A9 85       LDA #$85
1213 8D 6D 45    STA $456D
1216 A9 86       LDA #$86
1218 8D 73 45    STA $4573
121B A9 87       LDA #$87
121D 8D 7D 45    STA $457D
1220 A9 89       LDA #$89
1222 8D 84 45    STA $4584
1225 A9 8A       LDA #$8A
1227 8D 8A 45    STA $458A
122A A9 8B       LDA #$8B
122C 8D 8E 45    STA $458E
122F A9 8C       LDA #$8C
1231 8D 93 45    STA $4593
1234 A9 88       LDA #$88
1236 8D 98 45    STA $4598
1239 60          RTS
```

In modo da realizzare la finestra di comando in *reverse* rispetto al testo (la routine di *interrupt* viene chiamata automaticamente dal sistema operativo del Commodore 16 ogni sessantesimo di secondo); e ridefinisce gli otto tasti funzione in modo da poterli utilizzare per ulteriori comandi. La nuova routine di *interrupt* è

```
1988 A0 28      LDY #$28
198A AE 2C 15   LDX $152C
198D B1 D0      LDA ($D0),Y
198F 09 80      ORA #$80
1991 91 D0      STA ($D0),Y
1993 8A         TXA
1994 91 D3      STA ($D3),Y
1996 88         DEY
1997 D0 F4      BNE $198D
1999 4C 0E CE   JMP $CE0E
```

che alla fine si ricollega alla normale routine ($CE0E). INIT2 quindi continua stampando ($11F6) la scritta 'Easyword' e quindi il nome dell'autore, che verrà cancellato alla prima pressione di un qualsiasi tasto. Alla fine un salto alla routine CONTROLLA pone termine a questa fondamentale subroutine.

```
11F6 20 4E 12   JSR $124E
11F9 A9 12      LDA #$12
11FB A0 26      LDY #$26
11FD 20 71 11   JSR $1171
1200 A9 00      LDA #$00
1202 8D 13 28   STA $2813
1205 60         RTS
```

Stampa, sulla finestra di comando, il messaggio 'Easyword' dopo aver chiamato **PULISCI**

```
124E A2 27      LDX #$27
1250 A9 20      LDA #$20
1252 9D 00 0C   STA $0C00,X
1255 CA         DEX
1256 10 FA      BPL $1252
1258 A9 13      LDA #$13
125A 4C D2 FF   JMP $FFD2
```

che cancella tutta la linea di comando, evitando così sovrapposizioni di messaggi.

```
125D 48        PHA
125E 29 80     AND #$80
1260 4A        LSR
1261 85 41     STA $41
1263 68        PLA
1264 29 3F     AND #$3F
1266 05 41     ORA $41
1268 60        RTS
```

Questa routine converte i caratteri da ASCII Commodore in valori per le 'POKE' sullo schermo: infatti ad esempio la lettera 'a' ha ASCII 65 ma codice di schermo 1.

```
1269 A0 00     LDY #$00
126B 8C 71 2C  STY $2C71
126E B1 39     LDA ($39),Y
1270 85 D5     STA $D5
1272 A0 00     LDY #$00
1274 B1 39     LDA ($39),Y
1276 49 80     EOR #$80
1278 91 39     STA ($39),Y
127A AD 71 2C  LDA $2C71
127D 49 01     EOR #$01
127F 8D 71 2C  STA $2C71
1282 20 9E 10  JSR $109E
1285 20 E4 FF  JSR $FFE4
1288 D0 0D     BNE $1297
128A A5 A5     LDA $A5
128C 29 10     AND #$10
128E F0 F5     BEQ $1285
1290 A9 00     LDA #$00
1292 85 A5     STA $A5
1294 4C 72 12  JMP $1272
1297 AA        TAX
1298 A0 00     LDY #$00
129A A5 D5     LDA $D5
129C 91 39     STA ($39),Y
129E 8C 71 2C  STY $2C71
12A1 E0 5F     CPX #$5F
```

```
12A3 D0 0C      BNE $12B1
12A5 20 70 14   JSR $1470
12A8 A9 20      LDA #$20
12AA A0 00      LDY #$00
12AC 91 39      STA ($39),Y
12AE 4C 69 12   JMP $1269
12B1 AD 13 28   LDA $2813
12B4 F0 07      BEQ $12BD
12B6 8A         TXA
12B7 48         PHA
12B8 20 F6 11   JSR $11F6
12BB 68         PLA
12BC AA         TAX
12BD 8A         TXA
12BE C9 0D      CMP #$0D
12C0 D0 02      BNE $12C4
12C2 A2 5F      LDX #$5F
12C4 8A         TXA
12C5 29 7F      AND #$7F
12C7 C9 20      CMP #$20
12C9 90 4E      BCC $1319
12CB E0 A0      CPX #$A0
12CD D0 02      BNE $12D1
12CF A2 20      LDX #$20
12D1 8A         TXA
12D2 48         PHA
12D3 A0 00      LDY #$00
12D5 B1 39      LDA ($39),Y
12D7 C9 1F      CMP #$1F
12D9 F0 05      BEQ $12E0
12DB AD 14 28   LDA $2814
12DE F0 03      BEQ $12E3
12E0 20 38 18   JSR $1838
12E3 68         PLA
12E4 20 5D 12   JSR $125D
12E7 A0 00      LDY #$00
12E9 91 39      STA ($39),Y
12EB 20 9E 10   JSR $109E
12EE 38         SEC
12EF A5 39      LDA $39
```

```
12F1 ED 17 28 SBC $2817
12F4 85 41    STA $41
12F6 A5 3A    LDA $3A
12F8 ED 18 28 SBC $2818
12FB 05 41    ORA $41
12FD 90 0E    BCC $130D
12FF A5 39    LDA $39
1301 69 00    ADC #$00
1303 8D 17 28 STA $2817
1306 A5 3A    LDA $3A
1308 69 00    ADC #$00
130A 8D 18 28 STA $2818
130D E6 39    INC $39
130F D0 02    BNE $1313
1311 E6 3A    INC $3A
1313 20 B1 13 JSR $13B1
1316 4C 69 12 JMP $1269
```

Questo sopra è il **MAIN**, ciò il nucleo centrale del programma: come potete vedere al termine c'è un'istruzione di salto nuovamente all'inizio. Il MAIN fa lampeggiare il cursore ($1269-$127F) e rimane in attesa della pressione di un tasto. Quindi (da $12D1) converte il carattere in codice per lo schermo e lo scrive nel testo, controllando se è attivo il modo inserimento. Nel caso sia stato invece premuto un carattere di controllo, viene attivato **CONTROLLO**

```
1319 8A       TXA
131A AE 3B 13 LDX $133B
131D DD 3B 13 CMP $133B,X
1320 F0 06    BEQ $1328
1322 CA       DEX
1323 D0 F8    BNE $131D
1325 4C 69 12 JMP $1269
1328 CA       DEX
1329 8A       TXA
132A 0A       ASL
132B AA       TAX
132C A9 12    LDA #$12
132E 48       PHA
132F A9 68    LDA #$68
```

```
1331 48       PHA
1332 BD 64 13 LDA $1364,X
1335 48       PHA
1336 BD 63 13 LDA $1363,X
1339 48       PHA
133A 60       RTS
```

che confronta il carattere ottenuto con quelli (39 in tutto) contenuti nella tabella da $133C a $1362. Nel caso di confronto positivo, il controllo viene passato alla routine che opera l'azione richiesta. Ciò viene realizzato con una particolare tecnica che realizza una sorta di 'ON... GOSUB': tra $1363 e $13BO sono contenuti gli indirizzi di memoria di partenza delle 39 routine, e quelli della routine corrispondente al carattere di controllo premuto sono forzati nello stack (da $1328 a $133A): in tal modo quando CONTROLLO termina ocn il RTS ($133A) il controllo passa alla subroutine desiderata. Quando anch'essa ha termine, si torna al MAIN.

```
13B1 20 0F 14 JSR $140F
13B4 38       SEC
13B5 A5 39    LDA $39
13B7 ED 11 28 SBC $2811
13BA A5 3A    LDA $3A
13BC ED 12 28 SBC $2812
13BF B0 20    BCS $13E1
13C1 38       SEC
13C2 AD 11 28 LDA $2811
13C5 ED 08 28 SBC $2808
13C8 85 41    STA $41
13CA AD 12 28 LDA $2812
13CD ED 09 28 SBC $2809
13D0 05 41    ORA $41
13D2 F0 0D    BEQ $13E1
13D4 A5 39    LDA $39
13D6 8D 11 28 STA $2811
13D9 A5 3A    LDA $3A
13DB 8D 12 28 STA $2812
13DE 20 9E 10 JSR $109E
13E1 38       SEC
13E2 AD 1B 28 LDA $281B
```

```
13E5 E5 39    SBC $39
13E7 85 3F    STA $3F
13E9 AD 1C 28 LDA $281C
13EC E5 3A    SBC $3A
13EE 85 40    STA $40
13F0 05 3F    ORA $3F
13F2 F0 02    BEQ $13F6
13F4 B0 18    BCS $140E
13F6 18       CLC
13F7 AD 11 28 LDA $2811
13FA 6D 10 28 ADC $2810
13FD 8D 11 28 STA $2811
1400 AD 12 28 LDA $2812
1403 69 00    ADC #$00
1405 8D 12 28 STA $2812
1408 20 9E 10 JSR $109E
140B 4C E1 13 JMP $13E1
140E 60       RTS
140F 38       SEC
1410 AD 17 28 LDA $2817
1413 ED 0A 28 SBC $280A
1416 85 41    STA $41
1418 AD 18 28 LDA $2818
141B ED 0B 28 SBC $280B
141E 05 41    ORA $41
1420 90 0C    BCC $142E
1422 AD 0A 28 LDA $280A
1425 8D 17 28 STA $2817
1428 AD 0B 28 LDA $280B
142B 8D 18 28 STA $2818
142E 38       SEC
142F A5 39    LDA $39
1431 ED 08 28 SBC $2808
1434 85 41    STA $41
1436 A5 3A    LDA $3A
1438 ED 09 28 SBC $2809
143B 05 41    ORA $41
143D B0 0B    BCS $144A
143F AD 08 28 LDA $2808
```

```
1442 85 39    STA $39
1444 AD 09 28 LDA $2809
1447 85 3A    STA $3A
1449 60       RTS
144A 38       SEC
144B A5 39    LDA $39
144D ED 17 28 SBC $2817
1450 85 41    STA $41
1452 A5 3A    LDA $3A
1454 ED 18 28 SBC $2818
1457 05 41    ORA $41
1459 B0 01    BCS $145C
145B 60       RTS
145C AD 17 28 LDA $2817
145F 85 39    STA $39
1461 AD 18 28 LDA $2818
1464 85 3A    STA $3A
1466 60       RTS
```

La routine **CONTROLLA**, chiamata al termine del MAIN, opera insieme a REGOLA per verificare la correttezza delle azioni del MAIN. Per prima cosa ($140F-$1466) controlla che il cursore non vada oltre il termine della memoria: altrimenti impedisce ogni ulteriore inserimento. Quindi ($13B4-$140E) controlla la posizione del cursore all'interno dello schermo, per evitare che ne esca: i due JMP a REGOLA (in $13DE e $1408) permettono lo *scroll* dello schermo in alto o in basso, a seconda della posizione del cursore.
Con CONTROLLA termina la parte principale del programma, o meglio quella riguardante lo schema su cui operano poi tutte le altre funzioni permessa da EASYWORD.

```
1467 E6 39    INC $39
1469 D0 02    BNE $146D
146B E6 3A    INC $3A
146D 4C B1 13 JMP $13B1
```

DESTRA permette lo spostamento del cursore di una posizione verso destra. Notate come questa routine, analogamente a tutte le seguenti, non realizzi direttamente lo spostamento ma si limiti ad incrementare la variabile ($0039) che contiene la posizione del cursore. Saranno poi REGOLA e CONTROLLA a dare

effettivamente luogo allo spostamento.

```
1470 A5 39     LDA $39
1472 D0 02     BNE $1476
1474 C6 3A     DEC $3A
1476 C6 39     DEC $39
1478 4C B1 13  JMP $13B1
```

SINISTRA permette invece lo spostamento di una posizione verso sinistra del cursore. Anche lei termina con una chiamata a CONTROLLA.

```
147B A5 39     LDA $39
147D 85 3F     STA $3F
147F A5 3A     LDA $3A
1481 85 40     STA $40
1483 C6 40     DEC $40
1485 A0 FF     LDY #$FF
1487 B1 3F     LDA ($3F),Y
1489 C9 20     CMP #$20
148B F0 04     BEQ $1491
148D C9 1F     CMP #$1F
148F D0 03     BNE $1494
1491 88        DEY
1492 D0 F3     BNE $1487
1494 B1 3F     LDA ($3F),Y
1496 C9 20     CMP #$20
1498 F0 08     BEQ $14A2
149A C9 1F     CMP #$1F
149C F0 04     BEQ $14A2
149E 88        DEY
149F D0 F3     BNE $1494
14A1 60        RTS
14A2 38        SEC
14A3 98        TYA
14A4 65 3F     ADC $3F
14A6 85 39     STA $39
14A8 A5 40     LDA $40
14AA 69 00     ADC #$00
14AC 85 3A     STA $3A
14AE 4C B1 13  JMP $13B1
```

PAROLASIN permette di posizionare il cursore all'inizio della parola precedente quella in cui si trova il cursore. In pratica viene qui ricercato il primo spazio a sinistra.

```
14B1  A0  00      LDY  #$00
14B3  B1  39      LDA  ($39),Y
14B5  C9  20      CMP  #$20
14B7  F0  08      BEQ  $14C1
14B9  C9  1F      CMP  #$1F
14BB  F0  04      BEQ  $14C1
14BD  C8          INY
14BE  D0  F3      BNE  $14B3
14C0  60          RTS
14C1  C8          INY
14C2  D0  0B      BNE  $14CF
14C4  E6  3A      INC  $3A
14C6  A5  3A      LDA  $3A
14C8  CD  18  28  CMP  $2818
14CB  90  02      BCC  $14CF
14CD  D0  19      BNE  $14E8
14CF  B1  39      LDA  ($39),Y
14D1  C9  20      CMP  #$20
14D3  F0  EC      BEQ  $14C1
14D5  C9  1F      CMP  #$1F
14D7  F0  E8      BEQ  $14C1
```

PAROLADES consente invece lo spostamento del cursore all'inizio della parola successiva. Tutte queste ultime 4 routine, come del resto anche le altre che regolano lo spostamento del cursore, non controllano che il cursore non vada oltre il testo: a ciò pensa già REGOLA; ecco l'utilità di centralizzare in un'unica routine la stampa sullo schermo.

```
14D9  18          CLC
14DA  98          TYA
14DB  65  39      ADC  $39
14DD  85  39      STA  $39
14DF  A5  3A      LDA  $3A
14E1  69  00      ADC  #$00
14E3  85  3A      STA  $3A
14E5  4C  B1  13  JMP  $13B1
```

AGGY aggiunge alla posizione del cursore (varibile $0039; in assembler in realtà non esistono variabili ma solo locazioni di memoria) il valore contenuto nel registro Y per lo spostamento del cursore.

```
14E8 AD 17 28 LDA $2817
14EB 85 39    STA $39
14ED AD 18 28 LDA $2818
14F0 85 3A    STA $3A
14F2 4C B1 13 JMP $13B1
```

NONFIN viene chiamata quando, in fase di ricerca, una parola è più lunga di 255 caratteri.

```
14F5 A9 00    LDA #$00
14F7 8D 11 28 STA $2811
14FA AD 18 28 LDA $2818
14FD 38       SEC
14FE E9 04    SBC #$04
1500 CD 09 28 CMP $2809
1503 B0 03    BCS $1508
1505 AD 09 28 LDA $2809
1508 8D 12 28 STA $2812
150B 20 9E 10 JSR $109E
150E 4C E8 14 JMP $14E8
```

FINETESTO permette il posizionamento del cursore alla fine del testo. Se il puntatore di fine testo ($151C) indica un area già visibile sullo schermo, qui viene spostato il cursore (istruzione $1508), altrimenti la fine viene raggiunta posizionandosi 1K dietro alla fine del testo per fare in modo che sia poi CONTROLLA ad operare lo *scroll* fino a mostrare nuovamente il cursore. Questo modo operativo è necessario poiché sullo schermo possono trovare posto sia 960 caratteri che solo 24 (tutti simboli di 'RETURN') e quindi tornando indietro di 1K (cioè di 1024 caratteri) si è sicuri di portare il cursore 'sotto' lo schermo, facendo agire così lo *scroll* di CONTROLLA.

```
1511 EE 1D 15 INC $151D
1514 AD 1D 15 LDA $151D
1517 09 F0    ORA #$F0
1519 8D 1D 15 STA $151D
151C 60       RTS
```

BORDO permette il cambiamento del colore dello sfondo dello schermo, agendo sulla cella $151D. La routine REGOLA trasferisce poi il valore di $151D nelle corrispondenti per il colore dello sfondo e del bordo. L'istruzione $1517 permette di mantenere a 7 la luminosità: volendo diminuirla si devono porre valori tra #$00 e #$EØ nella locazione $1518.

```
151E EE 2C 15 INC $152C
1521 AD 2C 15 LDA $152C
1524 29 4F    AND #$4F
1526 8D 2C 15 STA $152C
1529 4C 9E 10 JMP $109E
```

LETTERE realizza invece il cambiamento del colore del testo, incrementando la variabile $152C (sarà poi REGOLA a operare il cambio di colore). Per impedire, continuando ad incrementare, che la variabile arrivi a cambiare anche la luminosità, è stata posta l'istruzione $1524. Se volete cambiare la luminosità (fissata a 4) dovete modificare il valore della locazione $1525, ponendo come valori (in esadecimale) #$1F, #$2F,... fino a #$7F per avere tutte e 7 le luminosità. Per realizzare ciò è sufficiente, in 'modo monitor', disassemblare la cella $1524 mediante il camando

D 1524 1524

posizionarsi sull'ultimo valore della riga (il #$4F) e quindi battere il nuovo valore desiderato. Infine bisogna uscire dal monitor (comando 'X') e, da BASIC, eseguire un normale 'SAVE' su nastro o disco.

```
152D A5 39    LDA $39
152F 85 3F    STA $3F
1531 A5 3A    LDA $3A
1533 85 40    STA $40
1535 C6 40    DEC $40
1537 A0 FF    LDY #$FF
1539 B1 3F    LDA ($3F),Y
153B C9 2E    CMP #$2E
153D F0 0C    BEQ $154B
153F C9 21    CMP #$21
1541 F0 08    BEQ $154B
1543 C9 3F    CMP #$3F
1545 F0 04    BEQ $154B
```

```
1547 C9 1F       CMP #$1F
1549 D0 04       BNE $154F
154B 88          DEY
154C D0 EB       BNE $1539
154E 60          RTS
154F B1 3F       LDA ($3F),Y
1551 C9 2E       CMP #$2E
1553 F0 1B       BEQ $1570
1555 C9 21       CMP #$21
1557 F0 17       BEQ $1570
1559 C9 3F       CMP #$3F
155B F0 13       BEQ $1570
155D C9 1F       CMP #$1F
155F F0 0F       BEQ $1570
1561 88          DEY
1562 D0 EB       BNE $154F
1564 C6 40       DEC $40
1566 A5 40       LDA $40
1568 CD 08 28    CMP $2808
156B B0 E2       BCS $154F
156D 4C 86 15    JMP $1586
1570 84 41       STY $41
1572 C6 41       DEC $41
1574 C8          INY
1575 F0 0A       BEQ $1581
1577 B1 3F       LDA ($3F),Y
1579 C9 20       CMP #$20
157B F0 F7       BEQ $1574
157D 88          DEY
157E 4C A2 14    JMP $14A2
1581 A4 41       LDY $41
1583 4C 4F 15    JMP $154F
1586 AD 08 28    LDA $2808
1589 85 39       STA $39
158B AD 09 28    LDA $2809
158E 85 3A       STA $3A
1590 4C B1 13    JMP $13B1
```

FRASESIN permette lo spostamento del cursore da una frase alla precedente o fino all'inizio di quella in cui si trova. La routine ricerca il primo punto fermo (#$2E), affermativo (#$21) o interrogativo (#$3F) presente. Quindi fa avanzare il cursore fino a che non ci sono più spazi, in modo da raggiungere il punto giusto nel quale deve trovarsi.

```
1593  A0  00      LDY  #$00
1595  B1  39      LDA  ($39),Y
1597  C9  2E      CMP  #$2E
1599  F0  1D      BEQ  $15B8
159B  C9  21      CMP  #$21
159D  F0  19      BEQ  $15B8
159F  C9  3F      CMP  #$3F
15A1  F0  15      BEQ  $15B8
15A3  C9  1F      CMP  #$1F
15A5  F0  11      BEQ  $15B8
15A7  C8          INY
15A8  D0  EB      BNE  $1595
15AA  E6  3A      INC  $3A
15AC  A5  3A      LDA  $3A
15AE  CD  18  28  CMP  $2818
15B1  F0  E2      BEQ  $1595
15B3  90  E0      BCC  $1595
15B5  4C  E8  14  JMP  $14E8
15B8  C8          INY
15B9  D0  0E      BNE  $15C9
15BB  E6  3A      INC  $3A
15BD  A5  3A      LDA  $3A
15BF  CD  18  28  CMP  $2818
15C2  90  05      BCC  $15C9
15C4  F0  03      BEQ  $15C9
15C6  4C  E8  14  JMP  $14E8
15C9  B1  39      LDA  ($39),Y
15CB  C9  20      CMP  #$20
15CD  F0  E9      BEQ  $15B8
15CF  C9  2E      CMP  #$2E
15D1  F0  E5      BEQ  $15B8
15D3  C9  21      CMP  #$21
15D5  F0  E1      BEQ  $15B8
```

```
15D7 C9 3F    CMP #$3F
15D9 F0 DD    BEQ $15B8
15DB C9 1F    CMP #$1F
15DD F0 D9    BEQ $15B8
15DF 4C D9 14 JMP $14D9
```

FRASEDES opera, del tutto analogamente alla routine precedente, in modo da portare il cursore all'inizio della frase successiva a quella in cui si trova. Anche qui si cerca la terminazione della frase per portare poi il cursore alla fine degli spazi che la seguono.

```
15E2 AD 0C 28 LDA $280C
15E5 8D 8C 28 STA $288C
15E8 AD 0D 28 LDA $280D
15EB 8D 8D 28 STA $288D
15EE 20 4E 12 JSR $124E
15F1 A9 3A    LDA #$3A
15F3 A0 26    LDY #$26
15F5 20 71 11 JSR $1171
15F8 A9 01    LDA #$01
15FA 8D 13 28 STA $2813
15FD 60       RTS
```

PULISCIBUFF è la routine che realizza lo svuotamento del contenuto del *buffer*, operazione che viene realizzata prima di una cancellazione in modo da permettere il recupero di testo. La cancellazione agisce in modo immediato: infatti esiste una variabile di inizio *buffer* ($280C-$280D) ed una di fine testo nel *buffer* ($280E-$280F): azzerando quest'ultima si realizza lo svuotamento senza ricorrere a routine di cancellazione in memoria. Al termine il messaggio relativo viene stampato sulla finestra di comando ($15F1-$15FA).

```
15FE 38       SEC
15FF A5 39    LDA $39
1601 ED 08 28 SBC $2808
1604 85 41    STA $41
1606 A5 3A    LDA $3A
1608 ED 09 28 SBC $2809
160B 05 41    ORA $41
160D D0 03    BNE $1612
160F 68       PLA
```

```
1610 68          PLA
1611 60          RTS
1612 A5 39       LDA $39
1614 85 DB       STA $DB
1616 A5 3A       LDA $3A
1618 85 DC       STA $DC
161A 60          RTS
161B 38          SEC
161C A5 39       LDA $39
161E 85 DD       STA $DD
1620 49 FF       EOR #$FF
1622 65 DB       ADC $DB
1624 8D 90 28    STA $2890
1627 A5 3A       LDA $3A
1629 85 DE       STA $DE
162B 49 FF       EOR #$FF
162D 65 DC       ADC $DC
162F 8D 91 28    STA $2891
1632 A5 DB       LDA $DB
1634 8D 92 28    STA $2892
1637 A5 DC       LDA $DC
1639 8D 93 28    STA $2893
163C A5 DD       LDA $DD
163E 8D 94 28    STA $2894
1641 85 DB       STA $DB
1643 A5 DE       LDA $DE
1645 8D 95 28    STA $2895
1648 85 DC       STA $DC
164A 38          SEC
164B AD 91 28    LDA $2891
164E 6D 8D 28    ADC $288D
1651 CD 0F 28    CMP $280F
1654 90 14       BCC $166A
1656 20 4E 12    JSR $124E
1659 A9 49       LDA #$49
165B A0 26       LDY #$26
165D 20 71 11    JSR $1171
1660 A9 01       LDA #$01
1662 8D 13 28    STA $2813
1665 A9 00       LDA #$00
```

```
1667 85 EF       STA $EF
1669 60          RTS
166A AD 8C 28    LDA $288C
166D 85 DD       STA $DD
166F AD 8D 28    LDA $288D
1672 85 DE       STA $DE
1674 AD 90 28    LDA $2890
1677 85 E0       STA $E0
1679 18          CLC
167A 6D 8C 28    ADC $288C
167D 8D 8C 28    STA $288C
1680 AD 91 28    LDA $2891
1683 85 E1       STA $E1
1685 6D 8D 28    ADC $288D
1688 8D 8D 28    STA $288D
168B 20 23 10    JSR $1023
168E AD 92 28    LDA $2892
1691 85 DB       STA $DB
1693 AD 93 28    LDA $2893
1696 85 DC       STA $DC
1698 AD 94 28    LDA $2894
169B 85 DD       STA $DD
169D AD 95 28    LDA $2895
16A0 85 DE       STA $DE
16A2 38          SEC
16A3 AD 17 28    LDA $2817
16A6 E5 DD       SBC $DD
16A8 85 E0       STA $E0
16AA AD 18 28    LDA $2818
16AD E5 DE       SBC $DE
16AF 85 E1       STA $E1
16B1 20 23 10    JSR $1023
16B4 38          SEC
16B5 AD 17 28    LDA $2817
16B8 ED 90 28    SBC $2890
16BB 8D 17 28    STA $2817
16BE AD 18 28    LDA $2818
16C1 ED 91 28    SBC $2891
16C4 8D 18 28    STA $2818
16C7 60          RTS
```

CANCELLA è un insieme di routine che realizzano l'eliminazione di testo, e sono attivate dai comandi CTRL-W e CTRL-E. Utilizzano MUOVISU per effettuare realmente la cancellazione. Inizialmente ($15FE-$1611) c'è un controllo riguardante la posizione del cursore rispetto all'inizio del testo (variabile questa contenuta in $2808). I parametri per la cancellaizone sono fissati da $161B a $1654: viene valutata la posizione del cursore e la nuova posizione che deve assumere, e il testo da eliminare viene salvato nel *buffer* per preservarlo da eventuali errori. Nel caso il *buffer* sia pieno o vada in *overflow* (cioè si riempia più di quanto possa contenere), viene segnalato l'errore ($1656-$1669). La cancellazione vera e propria avviene tra le istruzioni $166A e $16C7, con le due chiamate a MUOVISU (indirizzi $168B e $16B1). Da notare che le routine, a seconda che eseguono un CTRL-W o un CTRL-E, fanno in modo che la posizione del cursore sia il punto di partenza o di arrivo per la cancellazione.

```
16DA 20 FE 15 JSR $15FE
16DD 20 70 14 JSR $1470
16E0 20 1B 16 JSR $161B
16E3 38       SEC
16E4 AD 8C 28 LDA $288C
16E7 E9 01    SBC #$01
16E9 8D 8C 28 STA $288C
16EC AD 8D 28 LDA $288D
16EF E9 00    SBC #$00
16F1 8D 8D 28 STA $288D
16F4 60       RTS
```

CANCAR permette la cancellazione di un singolo carattere: notate la compattezza della routine che si appoggia a CANCELLA e a SINISTRA. Il carattere cancellato, che viene salvato da CANCELLA nel *buffer*, è eliminato da esso ($16E3-$16F1) sottraendo 1 al *buffer pointer*.

```
16F5 AD 43 05 LDA $0543
16F8 C9 05    CMP #$05
16FA D0 03    BNE $16FF
16FC 4C 7A 17 JMP $177A
16FF 20 67 14 JSR $1467
1702 20 FE 15 JSR $15FE
1705 20 70 14 JSR $1470
1708 20 1B 16 JSR $161B
170B 4C E3 16 JMP $16E3
```

TOGLI, così breve, realizza una doppia azione: CTRL-K ($16FC) e SH-CTRL-K ($16FF-$170B). La prima azione è realizzata dalla routine TOGLISP, spiegata in seguito, mentre la seconda è ottenuta da più subroutine: DESTRA, CANCELLA, SINISTRA e CANCAR.

```
170E 20 E2 15 JSR $15E2
1711 A9 02    LDA #$02
1713 85 DA    STA $DA
1715 20 4E 12 JSR $124E
1718 A9 55    LDA #$55
171A A0 26    LDY #$26
171C 20 71 11 JSR $1171
171F 20 82 11 JSR $1182
1722 48       PHA
1723 20 F6 11 JSR $11F6
1726 68       PLA
1727 29 BF    AND #$BF
1729 C9 16    CMP #$16
172B D0 09    BNE $1736
172D 20 FE 15 JSR $15FE
1730 20 7B 14 JSR $147B
1733 4C 1B 16 JMP $161B
1736 C9 06    CMP #$06
1738 D0 09    BNE $1743
173A 20 FE 15 JSR $15FE
173D 20 2D 15 JSR $152D
1740 4C 1B 16 JMP $161B
1743 C9 10    CMP #$10
1745 D0 09    BNE $1750
1747 20 FE 15 JSR $15FE
174A 20 02 19 JSR $1902
174D 4C 1B 16 JMP $161B
1750 60       RTS
```

CANCELLAW è attivata da CTRL-W e realizza la cancellazione di testo precedente al cursore appoggiandosi alla routine CANCELLA. Per prima cosa mostra il messaggio relativo ($170E-$171C), quindi attende la pressione di un tasto ($171F) e valuta se deve eliminare una parola ($1729-$1733), una frase ($1736-$1740) o un paragrafo ($1743-$174D). Come potete vedere, richiama la subrouti-

ne PAROLAS (in $1730), FRASESIN ($173D) e PARAGSIN ($174A, questa ultima è presentata in seguito).

```
1751 38          SEC
1752 A5 39       LDA $39
1754 ED 11 28    SBC $2811
1757 85 41       STA $41
1759 A5 3A       LDA $3A
175B ED 12 28    SBC $2812
175E 05 41       ORA $41
1760 F0 0B       BEQ $176D
1762 AD 11 28    LDA $2811
1765 85 39       STA $39
1767 AD 12 28    LDA $2812
176A 85 3A       STA $3A
176C 60          RTS
176D AD 08 28    LDA $2808
1770 85 39       STA $39
1772 AD 09 28    LDA $2809
1775 85 3A       STA $3A
1777 4C B1 13    JMP $13B1
```

La routine sopra presentata è **HOME**, che posiziona il cursore nella posizione in alto a sinistra sullo schermo ($1751-$176C). Nel caso il cursore si trovi già in quella posizione, esso viene posizionato all'inizio del testo ($176D-1777). Il JMP finale a CONTROLLA permette lo *scroll* in basso del testo fino a far riapparire il cursore.

```
177A A5 39       LDA $39
177C 85 3F       STA $3F
177E 85 DD       STA $DD
1780 A5 3A       LDA $3A
1782 85 40       STA $40
1784 85 DE       STA $DE
1786 A0 00       LDY #$00
1788 B1 3F       LDA ($3F),Y
178A C9 20       CMP #$20
178C D0 1E       BNE $17AC
```

```
178E C8          INY
178F D0 F7       BNE $1788
1791 A5 40       LDA $40
1793 CD 18 28    CMP $2818
1796 90 0F       BCC $17A7
1798 AD 17 28    LDA $2817
179B 85 3F       STA $3F
179D AD 18 28    LDA $2818
17A0 85 40       STA $40
17A2 A0 00       LDY #$00
17A4 4C AC 17    JMP $17AC
17A7 E6 FC       INC $FC
17A9 4C 88 17    JMP $1788
17AC 18          CLC
17AD 98          TYA
17AE 65 3F       ADC $3F
17B0 85 DB       STA $DB
17B2 A9 00       LDA #$00
17B4 65 40       ADC $40
17B6 85 DC       STA $DC
17B8 38          SEC
17B9 AD 17 28    LDA $2817
17BC E5 DD       SBC $DD
17BE 85 E0       STA $E0
17C0 AD 18 28    LDA $2818
17C3 E5 DE       SBC $DE
17C5 85 E1       STA $E1
17C7 38          SEC
17C8 A5 DB       LDA $DB
17CA E5 DD       SBC $DD
17CC 8D 90 28    STA $2890
17CF A5 DC       LDA $DC
17D1 E5 DE       SBC $DE
17D3 8D 91 28    STA $2891
17D6 20 23 10    JSR $1023
17D9 38          SEC
17DA AD 17 28    LDA $2817
17DD ED 90 28    SBC $2890
17E0 8D 17 28    STA $2817
```

```
17E3 AD 18 28 LDA $2818
17E6 ED 91 28 SBC $2891
17E9 8D 18 28 STA $2818
17EC 60       RTS
```

Ecco **TOGLISP**, chiamata da TOGLI: elimina tutti gli spazi successivi al cursore. La loro ricerca avviene tra $1788 e $178F: dal ciclo si esce solo quando si trova un carattere diverso dallo spazio. La cancellazione ha luogo a partire da $17AC, in particolare con la chiamata a MUOVISU ($17D6), dopo aver posizionato correttamente tutti i parametri. Alla fine vengono riposizionate le variabili di fine testo ($17D9-$17E9).

```
17ED A9 FF    LDA #$FF
17EF 8D A9 28 STA $28A9
17F2 4C 07 18 JMP $1807
17F5 A9 05    LDA #$05
17F7 8D A9 28 STA $28A9
17FA 20 07 18 JSR $1807
17FD B1 39    LDA ($39),Y
17FF C9 20    CMP #$20
1801 D0 01    BNE $1804
1803 C8       INY
1804 4C D9 14 JMP $14D9
1807 A9 00    LDA #$00
1809 8D AA 28 STA $28AA
180C 20 4E 18 JSR $184E
180F A9 20    LDA #$20
1811 AE A9 28 LDX $28A9
1814 A0 00    LDY #$00
1816 91 39    STA ($39),Y
1818 C8       INY
1819 CA       DEX
181A D0 FA    BNE $1816
181C 60       RTS
```

255SPAZI inserisce 255 spazi dopo il cursore (comando RUN). Essa memorizza in $28A9 gli spazi da inserire e quindi passa il controllo a $1807 che, richiamando a sua volta INSERBLOC (spiegata in seguito) realizza l'azione. In questo modo tra $17F5 e $1804 è inserita **5SPAZI**, utilizzata dalla routine seguente.

```
181D 20 38 18 JSR $1838
1820 20 38 18 JSR $1838
1823 A9 1F    LDA #$1F
1825 A0 00    LDY #$00
1827 91 39    STA ($39),Y
1829 C8       INY
182A 91 39    STA ($39),Y
182C 20 9E 10 JSR $109E
182F 20 67 14 JSR $1467
1832 20 67 14 JSR $1467
1835 4C F5 17 JMP $17F5
```

SHRET realizza il comando SH-RETURN: inserisce due spazi (le due chiamate alla subroutine INSCAR, presentata subito dopo), li riempie di caratteri di 'RETURN' (simbolo #$1F: è la freccia a sinistra) e quindi richiama 5SPAZI.

```
1838 A9 01    LDA #$01
183A 8D A9 28 STA $28A9
183D A9 00    LDA #$00
183F 8D AA 28 STA $28AA
1842 20 4E 18 JSR $184E
1845 A9 20    LDA #$20
1847 A0 00    LDY #$00
1849 91 39    STA ($39),Y
184B 4C B1 13 JMP $13B1
184E 18       CLC
```

Ecco **INSCAR** che inserisce un singolo carattere, spostando il cursore a destra. Utilizza INSERBLOC.

```
184F AD 17 28 LDA $2817
1852 6D A9 28 ADC $28A9
1855 AD 18 28 LDA $2818
1858 6D AA 28 ADC $28AA
185B CD 0B 28 CMP $280B
185E 90 05    BCC $1865
1860 68       PLA
1861 68       PLA
```

```
1862 4C 9D 18 JMP $189D
1865 18       CLC
1866 A5 39    LDA $39
1868 85 DB    STA $DB
186A 6D A9 28 ADC $28A9
186D 85 DD    STA $DD
186F A5 3A    LDA $3A
1871 85 DC    STA $DC
1873 6D AA 28 ADC $28AA
1876 85 DE    STA $DE
1878 38       SEC
1879 AD 17 28 LDA $2817
187C E5 DB    SBC $DB
187E 85 E0    STA $E0
1880 AD 18 28 LDA $2818
1883 E5 DC    SBC $DC
1885 85 E1    STA $E1
1887 20 60 10 JSR $1060
188A 18       CLC
188B AD 17 28 LDA $2817
188E 6D A9 28 ADC $28A9
1891 8D 17 28 STA $2817
1894 AD 18 28 LDA $2818
1897 6D AA 28 ADC $28AA
189A 8D 18 28 STA $2818
189D 60       RTS
```

INSERBLOC: inserisce tanti spazi quanti sono contenuti nella variabile $18A9. Come vedete opera solo su puntatori.

```
189E AD 14 28 LDA $2814
18A1 49 0E    EOR #$0E
18A3 4C 3A 12 JMP $123A

123A 8D 14 28 STA $2814
123D F0 08    BEQ $1247
123F A9 29    LDA #$29
1241 8D FA 11 STA $11FA
```

```
1244 4C F6 11 JMP $11F6
1247 A9 01    LDA #$01
1249 4C A7 1A JMP $1AA7
```

```
1AA7 8D 13 28 STA $2813
1AAA 20 4E 12 JSR $124E
1AAD A9 12    LDA #$12
1AAF 8D FA 11 STA $11FA
1AB2 4C C8 16 JMP $16C8
```

```
16C8 A9 10    LDA #$10
16CA A0 1E    LDY #$1E
16CC 4C 71 11 JMP $1171
```

Questa è la routine **INSER**, che gestisce il modo inserimento (CTRL-I). Essa è spezzettata poiché è stata modificata in ultimo e ha così trovato posto nei 'buchi' lasciati dal resto del programma. La locazione $2814 contiene la variabile che regola i due modi normale e inserimento (è un *flag*). Nel caso sia stato selezionato il modo inserimento ($123F-$1244) viene cambiato l'indirizzo del messaggio che EASYWORD stampa continuamente nella finestra di comando (da 'Easyword' in 'Modo inserimento': in questo modo si conosce sempre il modo operativo nel quale ci si trova), altrimenti viene stampato il messaggio di ritorno al modo normale, ristabilendo la scritta 'Easyword' quale messaggio originario nella prima linea dello schermo.

```
18A7 A9 64    LDA #$64
18A9 A0 26    LDY #$26
18AB 20 71 11 JSR $1171
18AE 20 9F FF JSR $FF9F
18B1 20 E4 FF JSR $FFE4
18B4 F0 F8    BEQ $18AE
18B6 C9 93    CMP #$93
18B8 F0 F4    BEQ $18AE
18BA 29 7F    AND #$7F
18BC C9 53    CMP #$53
18BE 60       RTS
```

SIONO viene chiamata quando il programma richiede all'utente una risposta del tipo si o no. Viene mostrato il messaggio relativo ($18A7-$18AB), quindi viene confrontato il codice del tasto premuto dall'utente con quello della lettera 's' ($18BC).

```
18BF A9 02       LDA #$02
18C1 85 DA       STA $DA
18C3 20 4E 12    JSR $124E
18C6 A9 7B       LDA #$7B
18C8 A0 26       LDY #$26
18CA 20 71 11    JSR $1171
18CD 20 A7 18    JSR $18A7
18D0 F0 03       BEQ $18D5
18D2 4C F6 11    JMP $11F6
18D5 A2 FA       LDX #$FA
18D7 9A          TXS
18D8 20 37 11    JSR $1137
18DB 20 C5 11    JSR $11C5
18DE 4C 69 12    JMP $1269
```

CLEAR cancella tutto il testo in memoria. Prima chiede la conferma ($18C3-$18CD), chiamando anche SIONO, quindi attiva CANCELLA ($18D8) per eliminare tutto il testo, INIT ($18DB) per reinizializzare alcuni parametri e infine torna il controllo al MAIN ($18DE).

```
18E1 A0 00       LDY #$00
18E3 B1 39       LDA ($39),Y
18E5 C9 1F       CMP #$1F
18E7 F0 11       BEQ $18FA
18E9 C8          INY
18EA D0 F7       BNE $18E3
18EC E6 3A       INC $3A
18EE A5 3A       LDA $3A
18F0 CD 18 28    CMP $2818
18F3 90 EE       BCC $18E3
18F5 F0 EC       BEQ $18E3
18F7 4C E8 14    JMP $14E8
18FA C8          INY
```

```
18FB D0 02    BNE $18FF
18FD E6 3A    INC $3A
18FF 4C D9 14 JMP $14D9
```

PARAGDES permette lo spostamento del cursore di un paragrafo verso destra.

```
1902 A5 39    LDA $39
1904 85 3F    STA $3F
1906 A5 3A    LDA $3A
1908 85 40    STA $40
190A C6 40    DEC $40
190C A0 FF    LDY #$FF
190E B1 3F    LDA ($3F),Y
1910 C9 1F    CMP #$1F
1912 F0 11    BEQ $1925
1914 88       DEY
1915 C0 FF    CPY #$FF
1917 D0 F5    BNE $190E
1919 C6 40    DEC $40
191B A5 40    LDA $40
191D CD 09 28 CMP $2809
1920 B0 EC    BCS $190E
1922 4C 86 15 JMP $1586
1925 38       SEC
1926 98       TYA
1927 65 3F    ADC $3F
1929 85 3F    STA $3F
192B A9 00    LDA #$00
192D 65 40    ADC $40
192F 85 40    STA $40
1931 38       SEC
1932 A5 3F    LDA $3F
1934 E5 39    SBC $39
1936 85 41    STA $41
1938 A5 40    LDA $40
193A E5 3A    SBC $3A
193C 05 41    ORA $41
193E D0 12    BNE $1952
```

```
1940 84 41    STY $41
1942 18       CLC
1943 A5 3F    LDA $3F
1945 E5 41    SBC $41
1947 85 3F    STA $3F
1949 A5 40    LDA $40
194B E9 00    SBC #$00
194D 85 40    STA $40
194F 4C 14 19 JMP $1914
1952 A5 3F    LDA $3F
1954 85 39    STA $39
1956 A5 40    LDA $40
1958 85 3A    STA $3A
195A 4C B1 13 JMP $13B1
```

PARAGSIN realizza invece lo spostamento del cursore di un paragrafo verso sinistra, cioè indietro nel testo. Questo due routine non si trovano assieme alle altre relative agli spostamenti del cursore perché sono state aggiunte in un secondo tempo.

```
19A4 AD 43 05 LDA $0543
19A7 29 01    AND #$01
19A9 D0 03    BNE $19AE
19AB 20 E2 15 JSR $15E2
19AE 20 4E 12 JSR $124E
19B1 A9 8A    LDA #$8A
19B3 A0 26    LDY #$26
19B5 20 71 11 JSR $1171
19B8 A0 00    LDY #$00
19BA B1 39    LDA ($39),Y
19BC 49 80    EOR #$80
19BE 91 39    STA ($39),Y
19C0 20 9E 10 JSR $109E
19C3 A0 00    LDY #$00
19C5 B1 39    LDA ($39),Y
19C7 49 80    EOR #$80
19C9 91 39    STA ($39),Y
19CB A9 02    LDA #$02
```

```
19CD 85 DA       STA $DA
19CF 20 82 11    JSR $1182
19D2 09 40       ORA #$40
19D4 C9 56       CMP #$56
19D6 D0 09       BNE $19E1
19D8 20 01 1A    JSR $1A01
19DB 20 B1 14    JSR $14B1
19DE 4C 10 1A    JMP $1A10
19E1 C9 46       CMP #$46
19E3 D0 09       BNE $19EE
19E5 20 01 1A    JSR $1A01
19E8 20 93 15    JSR $1593
19EB 4C 10 1A    JMP $1A10
19EE C9 50       CMP #$50
19F0 D0 09       BNE $19FB
19F2 20 01 1A    JSR $1A01
19F5 20 E1 18    JSR $18E1
19F8 4C 10 1A    JMP $1A10
19FB 20 B1 13    JSR $13B1
19FE 4C F6 11    JMP $11F6
1A01 A5 39       LDA $39
1A03 85 DD       STA $DD
1A05 8D 86 28    STA $2886
1A08 A5 3A       LDA $3A
1A0A 85 DE       STA $DE
1A0C 8D 87 28    STA $2887
1A0F 60          RTS
1A10 38          SEC
1A11 A5 39       LDA $39
1A13 85 DB       STA $DB
1A15 ED 86 28    SBC $2886
1A18 8D 90 28    STA $2890
1A1B A5 3A       LDA $3A
1A1D 85 DC       STA $DC
1A1F ED 87 28    SBC $2887
1A22 8D 91 28    STA $2891
1A25 20 32 16    JSR $1632
1A28 AD 86 28    LDA $2886
1A2B 85 39       STA $39
```

```
1A2D AD 87 28 LDA $2887
1A30 85 3A    STA $3A
1A32 20 9E 10 JSR $109E
1A35 4C B8 19 JMP $19B8
```

CANCELLAE è attivata dal comando CTRL-E ed agisce in maniera del tutto analoga a CANCELLAW. Prima di tutto controlla se è premuto anche il tasto 'SHIFT' ($19A4-$19A7), nel qual caso non pulisce il *buffer*, quindi ($19AE-$19B5) stampa il messaggio relativo e rimane in attesa della pressione di un tasto che sia 'v' ($19D4), 'f' ($19E1) o 'p' ($1950). Le routine che operano le cancellazioni sono $1A01-$1A0F e $1A10-$1A35, la quale torna a $19B8 per permettere un'altra cancellazione.

```
1A38 A9 27    LDA #$27
1A3A E5 CA    SBC $CA
1A3C 8D 19 28 STA $2819
1A3F A0 00    LDY #$00
1A41 A9 5F    LDA #$5F
1A43 20 D2 FF JSR $FFD2
1A46 A9 9D    LDA #$9D
1A48 20 D2 FF JSR $FFD2
1A4B 8C 1A 28 STY $281A
1A4E 20 82 11 JSR $1182
1A51 AC 1A 28 LDY $281A
1A54 85 41    STA $41
1A56 A9 20    LDA #$20
1A58 20 D2 FF JSR $FFD2
1A5B A9 9D    LDA #$9D
1A5D 20 D2 FF JSR $FFD2
1A60 A9 9B    LDA #$9B
1A62 20 D2 FF JSR $FFD2
1A65 A5 41    LDA $41
1A67 C9 0D    CMP #$0D
1A69 F0 32    BEQ $1A9D
1A6B C9 14    CMP #$14
1A6D D0 0F    BNE $1A7E
1A6F 88       DEY
1A70 10 04    BPL $1A76
1A72 C8       INY
```

```
1A73 4C 41 1A JMP $1A41
1A76 A9 9D    LDA #$9D
1A78 20 D2 FF JSR $FFD2
1A7B 4C 41 1A JMP $1A41
1A7E A5 41    LDA $41
1A80 29 7F    AND #$7F
1A82 C9 20    CMP #$20
1A84 90 BB    BCC $1A41
1A86 CC 19 28 CPY $2819
1A89 F0 B6    BEQ $1A41
1A8B A5 41    LDA $41
1A8D 99 45 28 STA $2845,Y
1A90 20 D2 FF JSR $FFD2
1A93 A9 00    LDA #$00
1A95 85 CB    STA $CB
1A97 85 CF    STA $CF
1A99 C8       INY
1A9A 4C 41 1A JMP $1A41
1A9D 20 D2 FF JSR $FFD2
1AA0 A9 00    LDA #$00
1AA2 99 45 28 STA $2845,Y
1AA5 98       TYA
1AA6 60       RTS
```

INPUT è utilizzata per ottenere dall'utente nella finestra di comando un *input*, il quale viene memorizzato a partire dalla locazione $2845; $281A ne contiene la lunghezza. L'istruzione $1A4E richiede un carattere all'utente, e in seguito sono accettati soltanto i codici alfanumerici o la pressione del tasto DEL. La routine evita che il cursore, rappresentato dal simbolo della freccia a sinistra, esca dalla linea di comando.

```
1AB5 EA       NOP
1AB6 20 4E 12 JSR $124E
1AB9 A9 BC    LDA #$BC
1ABB A0 26    LDY #$26
1ABD 20 71 11 JSR $1171
1AC0 20 1C 1B JSR $1B1C
1AC3 B0 20    BCS $1AE5
```

```
1AC5 AD 08 28 LDA $2808
1AC8 85 3F    STA $3F
1ACA AD 09 28 LDA $2809
1ACD 85 40    STA $40
1ACF AE 17 28 LDX $2817
1AD2 AC 18 28 LDY $2818
1AD5 A9 3F    LDA #$3F
1AD7 20 D8 FF JSR $FFD8
1ADA B0 09    BCS $1AE5
1ADC A5 90    LDA $90
1ADE 29 BF    AND #$BF
1AE0 D0 03    BNE $1AE5
1AE2 4C 0A 1C JMP $1C0A
```

Con la routine **SAVE** iniziano alcune delle principali parti del programma che si occupano dei collegamenti con le periferiche: nastro, disco o stampante. La routine stampa il messaggio opportuno ($1AB5-$1ABD) e quindi richiama APRIFILE, trattata in seguito. Il salvataggio del testo avviene utilizzando la normale routine del sistema operativo (ecco perché i programmi salvati sono di tipo PRG, cioè richiamabili direttamente da BASIC, anche se non hanno la struttura di un normale programma BASIC): è l'istruzione $1AD7. Al termine ($1ADC-$1AE2), viene controllato l'eventuale errore (contenuto nella locazione $0090). Il salto finale è alla stampa del messaggio di 'ok'.

```
1AE5 F0 27    BEQ $1B0E
1AE7 AD 1B 1B LDA $1B1B
1AEA C9 08    CMP #$08
1AEC 90 06    BCC $1AF4
1AEE 20 96 23 JSR $2396
1AF1 4C 05 1B JMP $1B05
1AF4 AD 1B 1B LDA $1B1B
1AF7 C9 01    CMP #$01
1AF9 F0 F9    BEQ $1AF4
1AFB 20 4E 12 JSR $124E
1AFE A9 C2    LDA #$C2
1B00 A0 26    LDY #$26
1B02 20 71 11 JSR $1171
1B05 20 5D 19 JSR $195D
1B08 A9 01    LDA #$01
```

```
1B0A 8D 13 28 STA $2813
1B0D 60       RTS
1B0E 20 4E 12 JSR $124E
1B11 A9 CD    LDA #$CD
1B13 A0 26    LDY #$26
1B15 20 71 11 JSR $1171
1B18 4C 05 1B JMP $1B05
```

ERRORE è la routine che gestisce gli errori di input/output. In caso di pressione del tasto STOP ($1AE5 e $1B0E-$1B18) stampa l'opportuno messaggio; altrimenti guarda se la periferica era il drive (numero di canale 8), nel quale caso ne stampa l'errore riscontrato ($1AE7-$1AF1). Se invece l'errore di I/0 è avvenuto con il nastro, viene stampato un generico messaggio di errore ($1AF4-$1B0A).

```
1B1C 20 38 1A JSR $1A38
1B1F F0 16    BEQ $1B37
1B21 A9 EC    LDA #$EC
1B23 A0 26    LDY #$26
1B25 20 71 11 JSR $1171
1B28 20 82 11 JSR $1182
1B2B A2 08    LDX #$08
1B2D C9 44    CMP #$44
1B2F F0 0C    BEQ $1B3D
1B31 A2 01    LDX #$01
1B33 C9 4E    CMP #$4E
1B35 F0 06    BEQ $1B3D
1B37 20 F6 11 JSR $11F6
1B3A 68       PLA
1B3B 68       PLA
1B3C 60       RTS
1B3D 8E 1B 1B STX $1B1B
1B40 A9 01    LDA #$01
1B42 A0 00    LDY #$00
1B44 20 BA FF JSR $FFBA
1B47 A0 00    LDY #$00
1B49 E0 01    CPX #$01
1B4B F0 31    BEQ $1B7E
1B4D B9 45 28 LDA $2845,Y
1B50 C9 40    CMP #$40
```

```
1B52  D0  0E      BNE  $1B62
1B54  B9  46  28  LDA  $2846,Y
1B57  C9  3A      CMP  #$3A
1B59  F0  23      BEQ  $1B7E
1B5B  B9  47  28  LDA  $2847,Y
1B5E  C9  3A      CMP  #$3A
1B60  F0  1C      BEQ  $1B7E
1B62  A9  30      LDA  #$30
1B64  8D  6D  28  STA  $286D
1B67  A9  3A      LDA  #$3A
1B69  8D  6E  28  STA  $286E
1B6C  B9  45  28  LDA  $2845,Y
1B6F  99  6F  28  STA  $286F,Y
1B72  C8          INY
1B73  CC  1A  28  CPY  $281A
1B76  90  F4      BCC  $1B6C
1B78  F0  F2      BEQ  $1B6C
1B7A  C8          INY
1B7B  4C  8A  1B  JMP  $1B8A
1B7E  B9  45  28  LDA  $2845,Y
1B81  99  6D  28  STA  $286D,Y
1B84  C8          INY
1B85  CC  1A  28  CPY  $281A
1B88  D0  F4      BNE  $1B7E
1B8A  8C  85  28  STY  $2885
1B8D  20  4E  12  JSR  $124E
1B90  A9  45      LDA  #$45
1B92  A0  28      LDY  #$28
1B94  20  71  11  JSR  $1171
1B97  AD  85  28  LDA  $2885
1B9A  A2  6D      LDX  #$6D
1B9C  A0  28      LDY  #$28
1B9E  20  BD  FF  JSR  $FFBD
1BA1  A9  0D      LDA  #$0D
1BA3  20  D2  FF  JSR  $FFD2
1BA6  4C  45  1C  JMP  $1C45
```

APRIFILE ottiene il nome del *file* ($1B1C) e chiede di specificare la periferica ($1B1F-$1B35): nastro o disco. Definisce 1 come numero logico del canale

($1B40-$1B44) e quindi apre il *file*. Nel caso si utilizzi un'unità a dischi, aggiunge al nome del *file* i caratteri '0:' in modo da permettere il salvataggio per *default* dei testi sul drive 0 (ciò è utile solo in presenza di due drive). Ciò non avviene se il nome originario del file dato dall'utente inizi già per '0:' o '@0:'.

```
1BA9 20 4E 12 JSR $124E
1BAC A9 AA    LDA #$AA
1BAE A0 26    LDY #$26
1BB0 20 71 11 JSR $1171
1BB3 20 82 11 JSR $1182
1BB6 20 5D 12 JSR $125D
1BB9 09 80    ORA #$80
1BBB 48       PHA
1BBC AD 14 28 LDA $2814
1BBF F0 03    BEQ $1BC4
1BC1 20 38 18 JSR $1838
1BC4 20 F6 11 JSR $11F6
1BC7 68       PLA
1BC8 4C E7 12 JMP $12E7
```

FORMATO permette l'inserimento all'interno del tasto dei codici di formato, mostrando l'opportuno messaggio ($1BA9-$1BB0), rimanendo in attesa della pressione di un tasto ($1BB3), traducendolo in codici per lo schermo ($1BB6) e infine inserendolo nel testo.

```
1BCB 38       SEC
1BCC A5 39    LDA $39
1BCE ED 08 28 SBC $2808
1BD1 85 41    STA $41
1BD3 A5 3A    LDA $3A
1BD5 ED 09 28 SBC $2809
1BD8 05 41    ORA $41
1BDA F0 04    BEQ $1BE0
1BDC A9 05    LDA #$05
1BDE 85 DA    STA $DA
1BE0 20 4E 12 JSR $124E
1BE3 A9 00    LDA #$00
1BE5 A0 27    LDY #$27
1BE7 20 71 11 JSR $1171
```

```
1BEA 20 1C 1B JSR $1B1C
1BED A5 DA    LDA $DA
1BEF C9 05    CMP #$05
1BF1 F0 03    BEQ $1BF6
1BF3 20 37 11 JSR $1137
1BF6 A9 00    LDA #$00
1BF8 A6 39    LDX $39
1BFA A4 3A    LDY $3A
1BFC 20 D5 FF JSR $FFD5
1BFF 90 03    BCC $1C04
1C01 4C E5 1A JMP $1AE5
1C04 8E 17 28 STX $2817
1C07 8C 18 28 STY $2818
1C0A 20 E7 FF JSR $FFE7
1C0D 20 4E 12 JSR $124E
1C10 A9 E2    LDA #$E2
1C12 A0 26    LDY #$26
1C14 20 71 11 JSR $1171
1C17 4C 05 1B JMP $1B05
```

LOAD è la routine che permette il caricamento in memoria di *file* di testo. Per prima cosa viene controllata la posizione del cursore: se è all'inizio dell'area testo, viene eseguita una cancellazione del testo eventualmente già presente (CANCELLA, $1BF3). Viene stampato il messaggio di richiesta del nome del *file* ($1BE0-$1BE7) e il controllo passa ad APRIFILE ($1BEA). Il caricamento avviene in $1BFC utilizzando la routine del sistema operativo, e quindi o viene stampato un messaggio di operazione finita correttamente ($1C0D-$1C14) oppure si passa alla gestione degli errori (ERRORE, $1C01).

```
1C1A 20 4E 12 JSR $124E
1C1D A9 06    LDA #$06
1C1F A0 27    LDY #$27
1C21 20 71 11 JSR $1171
1C24 20 1C 1B JSR $1B1C
1C27 A9 01    LDA #$01
1C29 AE 08 28 LDX $2808
1C2C AC 09 28 LDY $2809
1C2F 20 D5 FF JSR $FFD5
1C32 A5 90    LDA $90
```

```
1C34 29 BF      AND #$BF
1C36 F0 D2      BEQ $1C0A
1C38 20 4E 12   JSR $124E
1C3B A9 D5      LDA #$D5
1C3D A0 26      LDY #$26
1C3F 20 71 11   JSR $1171
1C42 4C 05 1B   JMP $1B05
```

VERIFY espleta la funzione di verificare il salvataggio del testo presente in memoria. Chiede il nome del *file* ($1C1A-$1C21), attiva APRIFILE ($1C24), esegue la verifica ($1C27-$1C2F) e infine stampa, in caso di errore, il relativo messaggio ($1C38-$1C3F).

```
1C45 78         SEI
1C46 A9 F0      LDA #$F0
1C48 8D 15 FF   STA $FF15
1C4B 8D 19 FF   STA $FF19
1C4E A9 0E      LDA #$0E
1C50 8D 14 03   STA $0314
1C53 A9 CE      LDA #$CE
1C55 8D 15 03   STA $0315
1C58 A9 4D      LDA #$4D
1C5A 8D 3B 05   STA $053B
1C5D 58         CLI
1C5E 60         RTS
```

NORMALE permette il ritorno alla normale routine di *interrupt*, cosa necessaria prima di ogni operazione di input/output. Il colore dello schermo viene fissato nero, con scritte azzurre.

```
1C62 20 45 1C   JSR $1C45
1C65 A9 93      LDA #$93
1C67 20 D2 FF   JSR $FFD2
1C6A A9 0D      LDA #$0D
1C6C 20 D2 FF   JSR $FFD2
1C6F 20 94 1C   JSR $1C94
1C72 A9 0D      LDA #$0D
1C74 20 D2 FF   JSR $FFD2
```

```
1C77 A9 0E     LDA #$0E
1C79 A0 27     LDY #$27
1C7B 20 71 11  JSR $1171
1C7E 20 E4 FF  JSR $FFE4
1C81 C9 0D     CMP #$0D
1C83 D0 F9     BNE $1C7E
1C85 20 5D 19  JSR $195D
1C88 4C F6 11  JMP $11F6
1C8B 20 CC FF  JSR $FFCC
1C8E A9 01     LDA #$01
1C90 20 C3 FF  JSR $FFC3
1C93 60        RTS
1C94 20 E7 FF  JSR $FFE7
1C97 A9 01     LDA #$01
1C99 A2 08     LDX #$08
1C9B A0 00     LDY #$00
1C9D 20 BA FF  JSR $FFBA
1CA0 A9 01     LDA #$01
1CA2 A2 43     LDX #$43
1CA4 A0 27     LDY #$27
1CA6 20 BD FF  JSR $FFBD
1CA9 20 C0 FF  JSR $FFC0
1CAC B0 DD     BCS $1C8B
1CAE A2 01     LDX #$01
1CB0 20 C6 FF  JSR $FFC6
1CB3 20 01 1D  JSR $1D01
1CB6 20 01 1D  JSR $1D01
1CB9 20 01 1D  JSR $1D01
1CBC 20 01 1D  JSR $1D01
1CBF F0 CA     BEQ $1C8B
1CC1 20 CC FF  JSR $FFCC
1CC4 20 E4 FF  JSR $FFE4
1CC7 C9 20     CMP #$20
1CC9 D0 03     BNE $1CCE
1CCB 20 82 11  JSR $1182
1CCE A2 01     LDX #$01
1CD0 20 C6 FF  JSR $FFC6
1CD3 20 01 1D  JSR $1D01
1CD6 48        PHA
```

```
1CD7 20 01 1D JSR $1D01
1CDA A8       TAY
1CDB 68       PLA
1CDC AA       TAX
1CDD 98       TYA
1CDE EA       NOP
1CDF EA       NOP
1CE0 EA       NOP
1CE1 EA       NOP
1CE2 20 5F A4 JSR $A45F
1CE5 EA       NOP
1CE6 EA       NOP
1CE7 EA       NOP
1CE8 EA       NOP
1CE9 A9 20    LDA #$20
1CEB 20 D2 FF JSR $FFD2
1CEE 20 01 1D JSR $1D01
1CF1 F0 06    BEQ $1CF9
1CF3 20 D2 FF JSR $FFD2
1CF6 4C EE 1C JMP $1CEE
1CF9 A9 0D    LDA #$0D
1CFB 20 D2 FF JSR $FFD2
1CFE 4C B9 1C JMP $1CB9
1D01 20 CF FF JSR $FFCF
1D04 48       PHA
1D05 A5 90    LDA $90
1D07 29 BF    AND #$BF
1D09 F0 06    BEQ $1D11
1D0B 68       PLA
1D0C 68       PLA
1D0D 68       PLA
1D0E 4C 8B 1C JMP $1C8B
1D11 68       PLA
1D12 60       RTS
```

CATALOGO permette di ottenere la stampa dei *file* contenuti sul dischetto senza sovrapporsi al testo contenuto in memoria. Disabilitato l'*interrupt* modificato e pulito lo schermo ($1C62-$1C6C), salta alla routine che si occupa della stampa a video ($0C94-$1D12, si comporta proprio come la routine di LIST del BASIC)

e rimane in attesa della pressione del tasto 'RETURN' ($1C7E-$1C83). Premendo la barra spaziatrice durante la stampa della *directory* del dischetto, essa si interrompe fino alla pressione di un altro tasto ($1CC4-$1CCB).

```
1D13 A2 00     LDX #$00
1D15 8E 88 28  STX $2888
1D18 8E 89 28  STX $2889
1D1B 8E 8A 28  STX $288A
1D1E 8E 8B 28  STX $288B
1D21 38        SEC
1D22 B1 3F     LDA ($3F),Y
1D24 E9 30     SBC #$30
1D26 90 2A     BCC $1D52
1D28 C9 0A     CMP #$0A
1D2A B0 26     BCS $1D52
1D2C 0E 88 28  ASL $2888
1D2F 2E 89 28  ROL $2889
1D32 0E 88 28  ASL $2888
1D35 2E 89 28  ROL $2889
1D38 0E 88 28  ASL $2888
1D3B 2E 89 28  ROL $2889
1D3E 0E 88 28  ASL $2888
1D41 2E 89 28  ROL $2889
1D44 0D 88 28  ORA $2888
1D47 8D 88 28  STA $2888
1D4A C8        INY
1D4B D0 D4     BNE $1D21
1D4D E6 40     INC $40
1D4F 4C 21 1D  JMP $1D21
1D52 F8        SED
1D53 AD 88 28  LDA $2888
1D56 0D 89 28  ORA $2889
1D59 F0 1C     BEQ $1D77
1D5B 38        SEC
1D5C AD 88 28  LDA $2888
1D5F E9 01     SBC #$01
1D61 8D 88 28  STA $2888
1D64 AD 89 28  LDA $2889
1D67 E9 00     SBC #$00
```

```
1D69 8D 89 28 STA $2889
1D6C EE 8A 28 INC $288A
1D6F D0 03    BNE $1D74
1D71 EE 8B 28 INC $288B
1D74 4C 53 1D JMP $1D53
1D77 AD 8A 28 LDA $288A
1D7A D8       CLD
1D7B 60       RTS
```

ASCIIHEX è la routine che traduce i codici di formato in effettivi comandi per la stampante, facendo in modo che il microprocessore lavori in BCD.

```
1D7C 38       SEC
1D7D AD 8C 28 LDA $288C
1D80 ED 0C 28 SBC $280C
1D83 8D 8E 28 STA $288E
1D86 AD 8D 28 LDA $288D
1D89 ED 0D 28 SBC $280D
1D8C 8D 8F 28 STA $288F
1D8F 0D 8E 28 ORA $288E
1D92 D0 10    BNE $1DA4
1D94 20 4E 12 JSR $124E
1D97 A9 34    LDA #$34
1D99 A0 27    LDY #$27
1D9B 20 71 11 JSR $1171
1D9E A9 01    LDA #$01
1DA0 8D 13 28 STA $2813
1DA3 60       RTS
1DA4 18       CLC
1DA5 A5 39    LDA $39
1DA7 85 DB    STA $DB
1DA9 6D 8E 28 ADC $288E
1DAC 85 DD    STA $DD
1DAE A5 3A    LDA $3A
1DB0 85 DC    STA $DC
1DB2 6D 8F 28 ADC $288F
1DB5 85 DE    STA $DE
1DB7 38       SEC
1DB8 AD 17 28 LDA $2817
```

```
1DBB E5 DB    SBC $DB
1DBD 85 E0    STA $E0
1DBF AD 18 28 LDA $2818
1DC2 E5 DC    SBC $DC
1DC4 85 E1    STA $E1
1DC6 18       CLC
1DC7 65 DE    ADC $DE
1DC9 CD 0B 28 CMP $280B
1DCC 90 10    BCC $1DDE
1DCE 20 4E 12 JSR $124E
1DD1 A9 27    LDA #$27
1DD3 A0 27    LDY #$27
1DD5 20 71 11 JSR $1171
1DD8 A9 01    LDA #$01
1DDA 8D 13 28 STA $2813
1DDD 60       RTS
1DDE 20 60 10 JSR $1060
1DE1 18       CLC
1DE2 AD 8E 28 LDA $288E
1DE5 85 E0    STA $E0
1DE7 6D 17 28 ADC $2817
1DEA 8D 17 28 STA $2817
1DED AD 8F 28 LDA $288F
1DF0 85 E1    STA $E1
1DF2 6D 18 28 ADC $2818
1DF5 8D 18 28 STA $2818
1DF8 A5 39    LDA $39
1DFA 85 DD    STA $DD
1DFC A5 3A    LDA $3A
1DFE 85 DE    STA $DE
1E00 AD 0C 28 LDA $280C
1E03 85 DB    STA $DB
1E05 AD 0D 28 LDA $280D
1E08 85 DC    STA $DC
1E0A 20 23 10 JSR $1023
1E0D 4C B1 13 JMP $13B1
```

RECBUF recupera il contenuto del *buffer* (comando CTRL-R), evitando che avvenga un *overflow* nel testo. Nel caso il *buffer* sia vuoto, stampa l'opportuno messaggio ($1D94-$1DA0), altrimenti verifica che il testo non superi la fine della memoria ($1DA4-$1DCC), nel cui caso stampa il relativo messaggio ($1DCE-$1DDA). Chiama MUOVIGIU per fare spazio in memoria ($1DDE) e, dopo aver fissato i parametri relativi in modo corretto, attiva MUOVISU ($1E0A) per trasferire il contenuto del *buffer* nel testo.

```
1E22 A0 00       LDY #$00
1E24 B1 39       LDA ($39),Y
1E26 AA          TAX
1E27 C8          INY
1E28 B1 39       LDA ($39),Y
1E2A 88          DEY
1E2B 91 39       STA ($39),Y
1E2D C8          INY
1E2E 8A          TXA
1E2F 91 39       STA ($39),Y
1E31 60          RTS
```

SCAMBIA inverte tra loro il carattere sotto il cursore con il successivo.

```
1E32 A0 00       LDY #$00
1E34 B1 39       LDA ($39),Y
1E36 29 3F       AND #$3F
1E38 F0 0A       BEQ $1E44
1E3A C9 1B       CMP #$1B
1E3C B0 06       BCS $1E44
1E3E B1 39       LDA ($39),Y
1E40 49 40       EOR #$40
1E42 91 39       STA ($39),Y
1E44 4C 67 14    JMP $1467
```

MINMAI converte il carattere sotto il cursore da minuscolo in maiuscolo e viceversa. Il cursore viene spostato di un posto a destra ($1E44) in modo da permettere il cambiamento rapido di intere frasi.

```
1E47 85 41       STA $41
1E49 29 3F       AND #$3F
1E4B 06 41       ASL $41
```

```
1E4D 24 41      BIT $41
1E4F 10 02      BPL $1E53
1E51 09 80      ORA #$80
1E53 70 02      BVS $1E57
1E55 09 40      ORA #$40
1E57 85 41      STA $41
1E59 60         RTS
```

ASCII converte i codici schermo dei caratteri in codici ASCII per la stampa.

```
1E6A 8D AF 28 STA $28AF
1E6D 8A       TXA
1E6E 48       PHA
1E6F 98       TYA
1E70 48       PHA
1E71 38       SEC
1E72 AD 9F 28 LDA $289F
1E75 ED A1 28 SBC $28A1
1E78 AD A0 28 LDA $28A0
1E7B ED A2 28 SBC $28A2
1E7E 90 1F    BCC $1E9F
1E80 AD AF 28 LDA $28AF
1E83 20 D2 FF JSR $FFD2
1E86 AD 43 05 LDA $0543
1E89 29 01    AND #$01
1E8B D0 F9    BNE $1E86
1E8D A5 91    LDA $91
1E8F C9 7F    CMP #$7F
1E91 D0 0C    BNE $1E9F
1E93 A5 91    LDA $91
1E95 C9 7F    CMP #$7F
1E97 F0 FA    BEQ $1E93
1E99 4C 78 20 JMP $2078
1E9C EA       NOP
1E9D EA       NOP
1E9E EA       NOP
1E9F 68       PLA
1EA0 A8       TAY
```

```
1EA1 68          PLA
1EA2 AA          TAX
1EA3 AD AF 28    LDA $28AF
1EA6 60          RTS
```

Questa è la prima delle routine di stampa. Tra $1E5A e $1E69 sono contenuti i valori di *default* dei codici di formato, unitamente a 4 caratteri per la stampante. Quella sopra è **STAMPACAR**, che realizza la stampa sulla periferica selezionata di un carattere ($1E83), tenendo conto di alcuni parametri quali ad esempio il numero della pagina. La pressione del tasto 'SHIFT' interrompe la stampa ($1E86-$1E8B), mentre quella del tasto 'STOP' la abortisce ($1E8D-$1E99).

```
1EA7 20 4E 12    JSR $124E
1EAA A9 A4       LDA #$A4
1EAC A0 27       LDY #$27
1EAE 4C 71 11    JMP $1171
1EB1 4C 78 20    JMP $2078
```

SCRMES scrive sullo schermo il messaggio che informa che la stampa è in atto.

```
1EB4 AD 1D 15    LDA $151D
1EB7 8D 6F 2C    STA $2C6F
1EBA AD 2C 15    LDA $152C
1EBD 85 DF       STA $DF
1EBF A9 4D       LDA #$4D
1EC1 20 E9 11    JSR $11E9
1EC4 20 BD FF    JSR $FFBD
1EC7 A9 04       LDA #$04
1EC9 8D AA 28    STA $28AA
1ECC A0 07       LDY #$07
1ECE AD 43 05    LDA $0543
1ED1 29 01       AND #$01
1ED3 D0 03       BNE $1ED8
1ED5 4C 68 1F    JMP $1F68
1ED8 20 4E 12    JSR $124E
1EDB A9 47       LDA #$47
1EDD A0 27       LDY #$27
```

```
1EDF  20  71  11  JSR  $1171
1EE2  20  82  11  JSR  $1182
1EE5  29  7F      AND  #$7F
1EE7  A2  03      LDX  #$03
1EE9  8E  AA  28  STX  $28AA
1EEC  C9  56      CMP  #$56
1EEE  F0  56      BEQ  $1F46
1EF0  A2  08      LDX  #$08
1EF2  8E  AA  28  STX  $28AA
1EF5  C9  44      CMP  #$44
1EF7  F0  22      BEQ  $1F1B
1EF9  C9  53      CMP  #$53
1EFB  D0  B4      BNE  $1EB1
1EFD  20  4E  12  JSR  $124E
1F00  A9  6D      LDA  #$6D
1F02  A0  27      LDY  #$27
1F04  20  71  11  JSR  $1171
1F07  20  82  11  JSR  $1182
1F0A  38          SEC
1F0B  E9  30      SBC  #$30
1F0D  C9  04      CMP  #$04
1F0F  90  A0      BCC  $1EB1
1F11  C9  50      CMP  #$50
1F13  B0  9C      BCS  $1EB1
1F15  8D  AA  28  STA  $28AA
1F18  4C  46  1F  JMP  $1F46
```

```
11E9  8D  2C  15  STA  $152C
11EC  A9  00      LDA  #$00
11EE  85  DA      STA  $DA
```

STAMPA è chiamata da CTRL-P. Dopo aver cambiato il colore dello schermo in nero ($1EB4-$1EC1 e poi REGOLA), viene aperto il canale 4 (la stampante) con indirizzo secondario 7 ($1EC7-$1ECC) a meno che non sia premuto anche il tasto 'SHIFT', nel quale caso chiede a quale periferica si vuole indirizzare la stampa: può essere il video ($1EEC, poi va a CONT), il disco ($1EF5, poi va a DISCO) o la stampante ($1EF9). In quest'ultimo caso chiede il numero della

periferica ($1EFD-$1F07) e va a CONT.

```
1F1B 20 4E 12 JSR $124E
1F1E A9 91    LDA #$91
1F20 A0 27    LDY #$27
1F22 20 71 11 JSR $1171
1F25 20 38 1A JSR $1A38
1F28 F0 87    BEQ $1EB1
1F2A AC 1A 28 LDY $281A
1F2D A9 2C    LDA #$2C
1F2F 99 45 28 STA $2845,Y
1F32 C8       INY
1F33 A9 57    LDA #$57
1F35 99 45 28 STA $2845,Y
1F38 C8       INY
1F39 8C 1A 28 STY $281A
1F3C AD 1A 28 LDA $281A
1F3F A2 45    LDX #$45
1F41 A0 28    LDY #$28
1F43 20 BD FF JSR $FFBD
```

DISCO chiede il nome del *file* di stampa, a cui aggiunge 'S,W,' poiché si tratta di un file di tipo sequenziale.

```
1F46 AD AA 28 LDA $28AA
1F49 A8       TAY
1F4A C9 04    CMP #$04
1F4C 90 1A    BCC $1F68
1F4E C9 08    CMP #$08
1F50 B0 16    BCS $1F68
1F52 20 4E 12 JSR $124E
1F55 A9 7B    LDA #$7B
1F57 A0 27    LDY #$27
1F59 20 71 11 JSR $1171
```

```
1F5C 20 82 11 JSR $1182
1F5F 38       SEC
1F60 E9 30    SBC #$30
1F62 A8       TAY
1F63 10 03    BPL $1F68
1F65 4C B1 1E JMP $1EB1
1F68 A9 01    LDA #$01
1F6A AE AA 28 LDX $28AA
1F6D 20 BA FF JSR $FFBA
1F70 20 A7 1E JSR $1EA7
1F73 A9 01    LDA #$01
1F75 20 C3 FF JSR $FFC3
1F78 20 C0 FF JSR $FFC0
1F7B A2 01    LDX #$01
1F7D 20 C9 FF JSR $FFC9
1F80 90 03    BCC $1F85
1F82 4C 78 20 JMP $2078
```

CONT continua la procedura di inizializzazione della periferica di output. Nel caso sia selezionata la stampante, chiede quale deve essere il numero secondario ($1F46-$1F5C). Quindi fissa a 1 il numero logico del file ($1F68-$1F6D), mostra la scritta che avverte della stampa in atto (SCRMES, $1F70) e finalmente da inizio alla stampa.

```
1F85 A2 00    LDX #$00
1F87 8E 97 28 STX $2897
1F8A 8E 96 28 STX $2896
1F8D 8E AB 28 STX $28AB
1F90 8E AC 28 STX $28AC
1F93 8E 70 2C STX $2C70
1F96 BD 5A 1E LDA $1E5A,X
1F99 9D 98 28 STA $2898,X
1F9C E8       INX
1F9D E0 0C    CPX #$0C
1F9F D0 F5    BNE $1F96
1FA1 A9 FF    LDA #$FF
1FA3 8D A6 28 STA $28A6
1FA6 8D A4 28 STA $28A4
1FA9 A2 04    LDX #$04
```

```
1FAB BD 65 1E LDA $1E65,X
1FAE 9D 1E 29 STA $291E,X
1FB1 CA       DEX
1FB2 D0 F7    BNE $1FAB
1FB4 AD 08 28 LDA $2808
1FB7 85 3F    STA $3F
1FB9 AD 09 28 LDA $2809
1FBC 85 40    STA $40
```

Questa routine inizializza alcuni *flag* di stampa ($1F85-$1F93): lunghezze del *footer* e dello *header*, stampa di ASCII standard, sottolineatura e *linefeed*. Quindi ricopia in opportune variabili alcuni valori di *default* per la stampante ($1F96-$1FB2) e infine fissa il punto di continuazione in caso di *file* collegati in fase di stampa ($1FB4-$1FBC).

```
1FBE A0 00    LDY #$00
1FC0 8C A5 28 STY $28A5
1FC3 CC A4 28 CPY $28A4
1FC6 F0 06    BEQ $1FCE
1FC8 AD 98 28 LDA $2898
1FCB 8D A5 28 STA $28A5
1FCE B1 3F    LDA ($3F),Y
1FD0 10 03    BPL $1FD5
1FD2 4C 62 21 JMP $2162
1FD5 C9 1F    CMP #$1F
1FD7 F0 2C    BEQ $2005
1FD9 99 6E 29 STA $296E,Y
1FDC C8       INY
1FDD EE A5 28 INC $28A5
1FE0 AD A5 28 LDA $28A5
1FE3 CD 99 28 CMP $2899
1FE6 90 E6    BCC $1FCE
1FE8 8C 16 28 STY $2816
1FEB B1 3F    LDA ($3F),Y
1FED C9 20    CMP #$20
1FEF F0 14    BEQ $2005
1FF1 CE A5 28 DEC $28A5
1FF4 88       DEY
1FF5 D0 F4    BNE $1FEB
```

```
1FF7 AC 16 28 LDY $2816
1FFA 4C 08 20 JMP $2008
1FFD C8       INY
1FFE B1 3F    LDA ($3F),Y
2000 C9 20    CMP #$20
2002 F0 01    BEQ $2005
2004 88       DEY
2005 8C 16 28 STY $2816
2008 98       TYA
2009 38       SEC
200A 65 3F    ADC $3F
200C 85 3F    STA $3F
200E A5 40    LDA $40
2010 69 00    ADC #$00
2012 85 40    STA $40
2014 A0 00    LDY #$00
2016 AD A6 28 LDA $28A6
2019 C9 FF    CMP #$FF
201B D0 03    BNE $2020
201D 20 09 21 JSR $2109
2020 AD A4 28 LDA $28A4
2023 F0 03    BEQ $2028
2025 20 31 21 JSR $2131
2028 38       SEC
2029 2E A4 28 ROL $28A4
202C AD 16 28 LDA $2816
202F 8D 15 28 STA $2815
2032 A9 6E    LDA #$6E
2034 85 E2    STA $E2
2036 A9 29    LDA #$29
2038 85 E3    STA $E3
203A 20 33 25 JSR $2533
```

MAINPRINT è il nucleo centrale che si occupa della stampa del testo in memoria, scrivendo il margine sinistro e facendo in modo che lo scritto non superi quello destro. Regola la stampa dello *header* ($2016-$201D) e infine effettua la stampa della linea ($2032-$203A).

```
203D 20 42 21 JSR $2142
2040 AD A6 28 LDA $28A6
2043 CD 9C 28 CMP $289C
2046 90 03    BCC $204B
2048 20 97 20 JSR $2097
204B 38       SEC
204C A5 3F    LDA $3F
204E ED 17 28 SBC $2817
2051 85 41    STA $41
2053 A5 40    LDA $40
2055 ED 18 28 SBC $2818
2058 05 41    ORA $41
205A F0 38    BEQ $2094
205C 90 36    BCC $2094
205E AD 97 28 LDA $2897
2061 F0 0B    BEQ $206E
2063 A9 00    LDA #$00
2065 8D 96 28 STA $2896
2068 8D 9B 28 STA $289B
206B 20 97 20 JSR $2097
206E AD AA 28 LDA $28AA
2071 C9 03    CMP #$03
2073 D0 03    BNE $2078
2075 20 82 11 JSR $1182
2078 20 E1 FF JSR $FFE1
207B F0 FB    BEQ $2078
207D A9 01    LDA #$01
207F 20 C3 FF JSR $FFC3
2082 20 E7 FF JSR $FFE7
2085 AD 6F 2C LDA $2C6F
2088 8D 1D 15 STA $151D
208B A2 FA    LDX #$FA
208D 9A       TXS
208E 20 F6 11 JSR $11F6
```

```
16CF A5 DF    LDA $DF
16D1 8D 2C 15 STA $152C
16D4 4C 69 12 JMP $1269
```

PROS prosegue la stampa: controlla se sia stata raggiunta la fine della pagina e stampa l'eventuale *footer* ($2048). Quindi guarda se ha raggiunto la fine del testo ($204B-$205C), nel qual caso guarda se deve stampare o meno il *footer*: in caso affermativo va a piè di pagina ($2063-$2097), altrimenti termina la stampa. Nel caso fosse selezionato lo schermo ($2063-$2075) rimane in attesa della pressione di un tasto prima di tornare in modo testo, quindi chiude tutti i *file* e va al MAIN dopo essere tornato ai colori selezionati per il testo e lo sfondo ($16CF-$16D4).

```
2097 38       SEC
2098 AD 9A 28 LDA $289A
209B ED A6 28 SBC $28A6
209E A8       TAY
209F 88       DEY
20A0 88       DEY
20A1 F0 08    BEQ $20AB
20A3 30 06    BMI $20AB
20A5 20 54 21 JSR $2154
20A8 88       DEY
20A9 D0 FA    BNE $20A5
20AB AD 97 28 LDA $2897
20AE F0 11    BEQ $20C1
20B0 8D 15 28 STA $2815
20B3 A9 6F    LDA #$6F
20B5 85 E2    STA $E2
20B7 A9 2B    LDA #$2B
20B9 85 E3    STA $E3
20BB 20 31 21 JSR $2131
20BE 20 33 25 JSR $2533
20C1 20 54 21 JSR $2154
20C4 20 54 21 JSR $2154
20C7 20 54 21 JSR $2154
20CA EE 9F 28 INC $289F
20CD D0 03    BNE $20D2
20CF EE A0 28 INC $28A0
20D2 AD 9E 28 LDA $289E
```

```
20D5 D0 32    BNE $2109
20D7 AD AA 28 LDA $28AA
20DA C9 03    CMP #$03
20DC F0 2B    BEQ $2109
20DE C9 08    CMP #$08
20E0 F0 27    BEQ $2109
20E2 38       SEC
20E3 AD 9F 28 LDA $289F
20E6 ED A1 28 SBC $28A1
20E9 AD A0 28 LDA $28A0
20EC ED A2 28 SBC $28A2
20EF 90 18    BCC $2109
20F1 20 CC FF JSR $FFCC
20F4 20 4E 12 JSR $124E
20F7 A9 B3    LDA #$B3
20F9 A0 27    LDY #$27
20FB 20 71 11 JSR $1171
20FE 20 82 11 JSR $1182
2101 20 A7 1E JSR $1EA7
2104 A2 01    LDX #$01
2106 20 C9 FF JSR $FFC9
```

IMPAGINA è la routine di impaginazione: arriva a fondo pagina e stampa o il *footer* ($20B3-$20BB) o una linea bianca, si posiziona su nuova pagina ($20CA-$20CF) e infine, nel caso la stampa non avvenga su video, rimane eventualmente in attesa della pressione di un tasto nel caso di stampa su fogli singoli ($20F4-$20FB).

```
2109 AD 96 28 LDA $2896
210C F0 11    BEQ $211F
210E 8D 15 28 STA $2815
2111 A9 6E    LDA #$6E
2113 85 E2    STA $E2
2115 A9 2A    LDA #$2A
2117 85 E3    STA $E3
2119 20 31 21 JSR $2131
211C 20 33 25 JSR $2533
211F AC 9B 28 LDY $289B
```

```
2122 8C A6 28 STY $28A6
2125 88       DEY
2126 F0 08    BEQ $2130
2128 30 06    BMI $2130
212A 20 54 21 JSR $2154
212D 88       DEY
212E D0 FA    BNE $212A
2130 60       RTS
```

CONT2 stampa lo *header* ($2111-$2119) e quindi si posiziona sulla prima riga del foglio su cui si può stampare ($211F-$212E).

```
2131 A9 20    LDA #$20
2133 AC 98 28 LDY $2898
2136 8C A5 28 STY $28A5
2139 F0 06    BEQ $2141
213B 20 6A 1E JSR $1E6A
213E 88       DEY
213F D0 FA    BNE $213B
2141 60       RTS
2142 AC 9D 28 LDY $289D
2145 18       CLC
2146 98       TYA
2147 6D A6 28 ADC $28A6
214A 8D A6 28 STA $28A6
214D 20 54 21 JSR $2154
2150 88       DEY
2151 D0 FA    BNE $214D
2153 60       RTS
2154 A9 0D    LDA #$0D
2156 20 6A 1E JSR $1E6A
2159 AD 70 2C LDA $2C70
215C F0 03    BEQ $2161
215E 20 6A 1E JSR $1E6A
2161 60       RTS
2162 8D A8 28 STA $28A8
2165 29 7F    AND #$7F
2167 20 47 1E JSR $1E47
```

```
216A AE AD 21 LDX $21AD
216D DD AD 21 CMP $21AD,X
2170 F0 09    BEQ $217B
2172 CA       DEX
2173 D0 F8    BNE $216D
2175 CE A5 28 DEC $28A5
2178 4C BE 22 JMP $22BE
217B CA       DEX
217C 8A       TXA
217D 0A       ASL
217E AA       TAX
217F 8C A7 28 STY $28A7
2182 A9 21    LDA #$21
2184 48       PHA
2185 A9 90    LDA #$90
2187 48       PHA
2188 BD C1 21 LDA $21C1,X
218B 48       PHA
218C BD C0 21 LDA $21C0,X
218F 48       PHA
2190 60       RTS
2191 38       SEC
2192 AD A7 28 LDA $28A7
2195 65 3F    ADC $3F
2197 85 3F    STA $3F
2199 A5 40    LDA $40
219B 69 00    ADC #$00
219D 85 40    STA $40
219F 4C BE 1F JMP $1FBE
21A2 B1 3F    LDA ($3F),Y
21A4 C9 1F    CMP #$1F
21A6 F0 01    BEQ $21A9
21A8 88       DEY
21A9 8C A7 28 STY $28A7
21AC 60       RTS
```

Sono queste alcune brevi routine di stampa. $2131-$2141 regola il margine sinistro; $2142-$2153 regola la spaziatura e tiene il conteggio delle linee stampate su ogni pagina; $2154-$2161 stampa una linea vuota ed eventualmente anche

un *linefeed*; $2162-$2190 controlla le marginature in accordo ai valori fissati dall'utente per eventuali centrature; $2191-$21AC evita la stampa di un codice di formato e dell'eventuale simbolo di 'RETURN' successivo.

```
21E4 C8          INY
21E5 A9 00       LDA #$00
21E7 8D A4 28    STA $28A4
21EA 4C A2 21    JMP $21A2
21ED C8          INY
21EE 20 13 1D    JSR $1D13
21F1 8D A3 28    STA $28A3
21F4 4C A2 21    JMP $21A2
21F7 C8          INY
21F8 20 13 1D    JSR $1D13
21FB 8D A1 28    STA $28A1
21FE AD 8B 28    LDA $288B
2201 8D A2 28    STA $28A2
2204 4C A2 21    JMP $21A2
2207 C8          INY
2208 20 13 1D    JSR $1D13
220B 8D 9F 28    STA $289F
220E AD 8B 28    LDA $288B
2211 8D A0 28    STA $28A0
2214 4C A2 21    JMP $21A2
2217 C8          INY
2218 20 13 1D    JSR $1D13
221B 8D 9A 28    STA $289A
221E 4C A2 21    JMP $21A2
2221 A9 00       LDA #$00
2223 8D 9E 28    STA $289E
2226 C8          INY
2227 4C A2 21    JMP $21A2
222A A9 0A       LDA #$0A
222C 8D 70 2C    STA $2C70
222F C8          INY
2230 4C A2 21    JMP $21A2
2233 C8          INY
2234 A9 01       LDA #$01
2236 8D AB 28    STA $28AB
```

```
2239 4C A2 21 JMP $21A2
223C C8       INY
223D 20 13 1D JSR $1D13
2240 8D 98 28 STA $2898
2243 4C A2 21 JMP $21A2
```

Sono riunite qui sopra 9 della 18 routine che permettono l'azione dei codici di formato, riconosciuti nella tabella $21AE-$21BF. Esse considerano i seguenti codici: m (marginatura, $21E4-$21EA); x (larghezza della pagina, $21ED-$21F4); ? (inizio stampa ad una pagina fissata, $21F7-$2204); @ (numerazione pagine, $2207-$2214); p (lunghezza pagina, $2217-$221E); w (attesa a fine pagina, $2221-$2227); j (*linefeed*, $222A-$2230); a (stampa in ASCII standard, $2233-$2239); l (margine sinistro, $223C-$2243).

```
2246 C8       INY
2247 20 13 1D JSR $1D13
224A 8D 99 28 STA $2899
224D 4C A2 21 JMP $21A2
2250 C8       INY
2251 20 13 1D JSR $1D13
2254 8D 9B 28 STA $289B
2257 4C A2 21 JMP $21A2
225A C8       INY
225B 20 13 1D JSR $1D13
225E 8D 9C 28 STA $289C
2261 4C A2 21 JMP $21A2
2264 C8       INY
2265 20 13 1D JSR $1D13
2268 8D 9D 28 STA $289D
226B 4C A2 21 JMP $21A2
226E AC A7 28 LDY $28A7
2271 C8       INY
2272 98       TYA
2273 48       PHA
2274 20 97 20 JSR $2097
2277 68       PLA
2278 A8       TAY
2279 8C A7 28 STY $28A7
227C 60       RTS
```

```
227D 20 97 22 JSR $2297
2280 88       DEY
2281 8C 96 28 STY $2896
2284 A0 01    LDY #$01
2286 B1 3F    LDA ($3F),Y
2288 99 6D 2A STA $2A6D,Y
228B C8       INY
228C CC 96 28 CPY $2896
228F 90 F5    BCC $2286
2291 F0 F3    BEQ $2286
2293 C8       INY
2294 4C A2 21 JMP $21A2
2297 C8       INY
2298 B1 3F    LDA ($3F),Y
229A C9 1F    CMP #$1F
229C D0 F9    BNE $2297
229E 60       RTS
229F 20 97 22 JSR $2297
22A2 88       DEY
22A3 8C 97 28 STY $2897
22A6 A0 01    LDY #$01
22A8 B1 3F    LDA ($3F),Y
22AA 99 6E 2B STA $2B6E,Y
22AD C8       INY
22AE CC 97 28 CPY $2897
22B1 90 F5    BCC $22A8
22B3 F0 F3    BEQ $22A8
22B5 4C A2 21 JMP $21A2
22B8 20 97 22 JSR $2297
22BB 4C A2 21 JMP $21A2
```

Ed ecco le altre 9 routine relative ai codici di formato: r (margine destro, $2246-$224D); t (margine superiore, $2250-$2257); b (margine inferiore, $225A-$2261); s (spaziatura interlineare, $2264-$226B); n (salto a nuova pagina, $226E-$227C); h (definisce lo *header*, $227D-$2294); salta al primo carattere dopo il successivo 'RETURN'($2294-$229E); f (definisce il *footer*, $229F-$22B5); i (ignora una linea di commento, $22B8-$22BB)

```
22BE C8        INY
22BF B1 3F     LDA ($3F),Y
22C1 C9 3D     CMP #$3D
22C3 F0 07     BEQ $22CC
22C5 88        DEY
22C6 AD A8 28  LDA $28A8
22C9 4C D9 1F  JMP $1FD9
22CC C8        INY
22CD 20 13 1D  JSR $1D13
22D0 48        PHA
22D1 AD A8 28  LDA $28A8
22D4 29 7F     AND #$7F
22D6 AA        TAX
22D7 68        PLA
22D8 9D EE 28  STA $28EE,X
22DB 20 A2 21  JSR $21A2
22DE 4C 91 21  JMP $2191
```

DEFINITI permette invece di gestire i codici di controllo definiti dall'utente. Ricerca il simbolo '=' ($22C1) dopo il codice. Se non viene trovato, il codice viene ignorato, altrimenti il valore successivo viene messo in coda al *buffer* di stampa ($22CC-$22DE).

```
22E1 C8        INY
22E2 A2 08     LDX #$08
22E4 B1 3F     LDA ($3F),Y
22E6 29 3F     AND #$3F
22E8 C9 04     CMP #$04
22EA F0 09     BEQ $22F5
22EC A2 01     LDX #$01
22EE C9 0E     CMP #$0E
22F0 F0 03     BEQ $22F5
22F2 4C B1 1E  JMP $1EB1
22F5 8E 1B 1B  STX $1B1B
22F8 C8        INY
22F9 B1 3F     LDA ($3F),Y
22FB C9 3A     CMP #$3A
22FD F0 03     BEQ $2302
22FF 4C B1 1E  JMP $1EB1
```

```
2302 C8          INY
2303 B1 3F       LDA ($3F),Y
2305 C9 1F       CMP #$1F
2307 F0 09       BEQ $2312
2309 20 47 1E    JSR $1E47
230C 99 6A 28    STA $286A,Y
230F 4C 02 23    JMP $2302
2312 98          TYA
2313 38          SEC
2314 E9 03       SBC #$03
2316 A2 6D       LDX #$6D
2318 A0 28       LDY #$28
231A 20 BD FF    JSR $FFBD
231D 20 CC FF    JSR $FFCC
2320 A9 02       LDA #$02
2322 20 C3 FF    JSR $FFC3
2325 A9 02       LDA #$02
2327 AE 1B 1B    LDX $1B1B
232A A0 00       LDY #$00
232C 20 BA FF    JSR $FFBA
232F 20 37 11    JSR $1137
2332 A9 00       LDA #$00
2334 A6 39       LDX $39
2336 A4 3A       LDY $3A
2338 20 D5 FF    JSR $FFD5
233B 90 03       BCC $2340
233D 4C B1 1E    JMP $1EB1
2340 8E 17 28    STX $2817
2343 8C 18 28    STY $2818
2346 68          PLA
2347 68          PLA
2348 A2 01       LDX #$01
234A 20 C9 FF    JSR $FFC9
234D 4C B4 1F    JMP $1FB4
```

CONCATENA permette di concatenare, in fase di stampa, un *file* ad un altro. Per prima cosa viene ricercata la periferica ($22E1-$22F2): ‘n’ o ‘d’ per nastro o disco. Dopo aver preparato le variabili fondamentali per mantenere le caratteristi-

che di stampa ($22F5-$230F), cancella la memoria-testo ($232F) e carica il *file* ($2338). In caso di operazione corretta, la stampa prosegue ($234D).

```
2350 20 E7 FF JSR $FFE7
2353 A9 00    LDA #$00
2355 20 BD FF JSR $FFBD
2358 A9 0F    LDA #$0F
235A A2 08    LDX #$08
235C A0 0F    LDY #$0F
235E 20 BA FF JSR $FFBA
2361 20 C0 FF JSR $FFC0
2364 90 0B    BCC $2371
2366 A9 0F    LDA #$0F
2368 20 C3 FF JSR $FFC3
236B 20 E7 FF JSR $FFE7
236E 4C F6 11 JMP $11F6
2371 20 4E 12 JSR $124E
2374 A9 1D    LDA #$1D
2376 A0 27    LDY #$27
2378 20 71 11 JSR $1171
237B 20 38 1A JSR $1A38
237E F0 16    BEQ $2396
2380 A2 0F    LDX #$0F
2382 20 C9 FF JSR $FFC9
2385 B0 DF    BCS $2366
2387 A9 45    LDA #$45
2389 A0 28    LDY #$28
238B 20 71 11 JSR $1171
238E A9 0D    LDA #$0D
2390 20 D2 FF JSR $FFD2
2393 20 CC FF JSR $FFCC
2396 20 E7 FF JSR $FFE7
2399 A9 00    LDA #$00
239B 20 BD FF JSR $FFBD
239E A9 0F    LDA #$0F
23A0 A2 08    LDX #$08
23A2 A0 0F    LDY #$0F
23A4 20 BA FF JSR $FFBA
23A7 20 C0 FF JSR $FFC0
```

```
23AA B0 BA       BCS $2366
23AC 20 4E 12    JSR $124E
23AF A2 0F       LDX #$0F
23B1 20 C6 FF    JSR $FFC6
23B4 20 38 1A    JSR $1A38
23B7 20 CC FF    JSR $FFCC
23BA A9 0F       LDA #$0F
23BC 20 C3 FF    JSR $FFC3
23BF 20 E7 FF    JSR $FFE7
23C2 A9 01       LDA #$01
23C4 8D 13 28    STA $2813
23C7 60          RTS
```

D'ora in poi, a partire da questa sopra, terminano le routine di stampa. **DISKCMD** permette di dare al *drive* un comando; se l'utente preme il solo 'RETURN', viene mostrato lo stato della periferica. Come potete vedere, il *drive* viene selezionato con l'indirizzo secondario 15 ($2380) che corrisponde al canale di comando. Il messaggio di ritorno del *drive* è mostrato in ogni caso ($2387-$238B).

```
23C8 20 F0 23    JSR $23F0
23CB AD B0 28    LDA $28B0
23CE F0 16       BEQ $23E6
23D0 20 93 24    JSR $2493
23D3 20 16 24    JSR $2416
23D6 AD AE 28    LDA $28AE
23D9 C9 FF       CMP #$FF
23DB F0 09       BEQ $23E6
23DD 20 B6 24    JSR $24B6
23E0 20 9E 10    JSR $109E
23E3 4C D3 23    JMP $23D3
23E6 4C F6 11    JMP $11F6
```

CERCSOS è la routine che opera l'azione di ricerca e di sostituzione automatica di testo all'interno del documento. La sua brevità è dovuta al fatto che continua a richiamare le routine, spiegate in seguito, CERCA ($23D3) e SOSTITUISCI ($23DD) fino a che CERCA segnala di non aver trovato nulla (locazione $28AE contenente #$FF).

```
23E9  AD  43  05  LDA  $0543
23EC  C9  05      CMP  #$05
23EE  D0  26      BNE  $2416
23F0  20  4E  12  JSR  $124E
23F3  A9  D5      LDA  #$D5
23F5  A0  27      LDY  #$27
23F7  20  71  11  JSR  $1171
23FA  20  38  1A  JSR  $1A38
23FD  8D  B0  28  STA  $28B0
2400  D0  03      BNE  $2405
2402  4C  F6  11  JMP  $11F6
2405  A0  00      LDY  #$00
2407  B9  45  28  LDA  $2845,Y
240A  99  B1  28  STA  $28B1,Y
240D  C8          INY
240E  CC  1A  28  CPY  $281A
2411  D0  F4      BNE  $2407
2413  4C  F6  11  JMP  $11F6
2416  A5  39      LDA  $39
2418  85  3F      STA  $3F
241A  A5  3A      LDA  $3A
241C  85  40      STA  $40
241E  A9  FF      LDA  #$FF
2420  8D  AE  28  STA  $28AE
2423  A0  01      LDY  #$01
2425  A2  00      LDX  #$00
2427  AD  B0  28  LDA  $28B0
242A  F0  50      BEQ  $247C
242C  BD  B1  28  LDA  $28B1,X
242F  20  5D  12  JSR  $125D
2432  D1  3F      CMP  ($3F),Y
2434  F0  02      BEQ  $2438
2436  A2  FF      LDX  #$FF
2438  C8          INY
2439  D0  0B      BNE  $2446
243B  E6  40      INC  $40
243D  A5  40      LDA  $40
243F  CD  18  28  CMP  $2818
2442  F0  02      BEQ  $2446
```

```
2444 B0 36    BCS $247C
2446 E8       INX
2447 EC B0 28 CPX $28B0
244A D0 E0    BNE $242C
244C 18       CLC
244D 98       TYA
244E 65 3F    ADC $3F
2450 85 41    STA $41
2452 A5 40    LDA $40
2454 69 00    ADC #$00
2456 85 42    STA $42
2458 AD 17 28 LDA $2817
245B C5 41    CMP $41
245D AD 18 28 LDA $2818
2460 E5 42    SBC $42
2462 90 18    BCC $247C
2464 38       SEC
2465 A5 41    LDA $41
2467 ED B0 28 SBC $28B0
246A 85 39    STA $39
246C 8D AD 28 STA $28AD
246F A5 42    LDA $42
2471 E9 00    SBC #$00
2473 85 3A    STA $3A
2475 8D AE 28 STA $28AE
2478 20 B1 13 JSR $13B1
247B 60       RTS
247C 20 4E 12 JSR $124E
247F A9 DF    LDA #$DF
2481 A0 27    LDY #$27
2483 20 71 11 JSR $1171
2486 A9 01    LDA #$01
2488 8D 13 28 STA $2813
248B 60       RTS
```

CERCA è la routine che permette il comando di ricerca. Se il tasto 'SHIFT' è premuto ($23E9-$23EE), viene richiesta ($23F0-$23FA) e memorizzata (a partire da $2845) la frase da ricercare. La ricerca avviene tra $2405 e $2411, e fallisce nel caso termini il testo ($242A), con la stampa del messaggio relativo ($247C-

$2488). La ricerca ha successo solo quando l'intera lunghezza (contenuta in $281A) della variabile $2845 è uguale al testo che si sta considerando: in questo caso ($2446-$247B) la posizione del cursore viene aggiornata e viene chiamata CONTROLLA ($2478) per effettuare lo *scroll* sullo schermo.

```
248C AD 43 05 LDA $0543
248F C9 05    CMP #$05
2491 D0 23    BNE $24B6
2493 20 4E 12 JSR $124E
2496 A9 E9    LDA #$E9
2498 A0 27    LDY #$27
249A 20 71 11 JSR $1171
249D 20 38 1A JSR $1A38
24A0 8D CF 28 STA $28CF
24A3 F0 0E    BEQ $24B3
24A5 A0 00    LDY #$00
24A7 B9 45 28 LDA $2845,Y
24AA 99 D0 28 STA $28D0,Y
24AD C8       INY
24AE CC 1A 28 CPY $281A
24B1 D0 F4    BNE $24A7
24B3 4C F6 11 JMP $11F6
24B6 38       SEC
24B7 A5 39    LDA $39
24B9 85 DD    STA $DD
24BB ED AD 28 SBC $28AD
24BE 85 41    STA $41
24C0 A5 3A    LDA $3A
24C2 85 DE    STA $DE
24C4 ED AE 28 SBC $28AE
24C7 05 41    ORA $41
24C9 D0 65    BNE $2530
24CB A9 FF    LDA #$FF
24CD 8D AE 28 STA $28AE
24D0 18       CLC
24D1 AD B0 28 LDA $28B0
24D4 65 39    ADC $39
24D6 85 DB    STA $DB
24D8 A9 00    LDA #$00
24DA 65 3A    ADC $3A
```

```
24DC 85 DC    STA $DC
24DE 38       SEC
24DF AD 17 28 LDA $2817
24E2 E5 DD    SBC $DD
24E4 85 E0    STA $E0
24E6 AD 18 28 LDA $2818
24E9 E5 DE    SBC $DE
24EB 85 E1    STA $E1
24ED 20 23 10 JSR $1023
24F0 38       SEC
24F1 AD 17 28 LDA $2817
24F4 ED B0 28 SBC $28B0
24F7 8D 17 28 STA $2817
24FA AD 18 28 LDA $2818
24FD E9 00    SBC #$00
24FF 8D 18 28 STA $2818
2502 AD CF 28 LDA $28CF
2505 F0 29    BEQ $2530
2507 8D A9 28 STA $28A9
250A A9 00    LDA #$00
250C 8D AA 28 STA $28AA
250F 20 4E 18 JSR $184E
2512 A0 00    LDY #$00
2514 B9 D0 28 LDA $28D0,Y
2517 20 5D 12 JSR $125D
251A 91 39    STA ($39),Y
251C C8       INY
251D CC CF 28 CPY $28CF
2520 D0 F2    BNE $2514
2522 18       CLC
2523 A5 39    LDA $39
2525 6D CF 28 ADC $28CF
2528 85 39    STA $39
252A A5 3A    LDA $3A
252C 69 00    ADC #$00
252E 85 3A    STA $3A
2530 4C B1 13 JMP $13B1
```

SOSTITUISCI effettua la sostituzione di testo. Se è premuto il tasto 'SHIFT' ($248C-$2491) viene richiesta e memorizzata la parola da sostituire ($2493-$24A0). Altrimenti la routine controlla se il cursore si trova nella posizione in cui era stato messo da CERCA, e solo dopo effettua la sostituzione del testo ($24B6-$250F). Il cursore viene al termine posizionato alla fine della parola sostituita ($2512-$252E), in modo che una successiva ricerca non possa andare in un ciclo senza fine.

```
2533 A0 00    LDY #$00
2535 CC 15 28 CPY $2815
2538 F0 20    BEQ $255A
253A B1 E2    LDA ($E2),Y
253C 30 1D    BMI $255B
253E 20 47 1E JSR $1E47
2541 20 D0 25 JSR $25D0
2544 20 6A 1E JSR $1E6A
2547 AD AC 28 LDA $28AC
254A F0 0A    BEQ $2556
254C A9 08    LDA #$08
254E 20 6A 1E JSR $1E6A
2551 A9 5F    LDA #$5F
2553 20 6A 1E JSR $1E6A
2556 C8       INY
2557 4C 35 25 JMP $2535
255A 60       RTS
255B 8C A7 28 STY $28A7
255E 29 7F    AND #$7F
2560 8D A8 28 STA $28A8
2563 20 47 1E JSR $1E47
2566 C9 43    CMP #$43
2568 D0 1B    BNE $2585
256A 38       SEC
256B AD A3 28 LDA $28A3
256E ED 15 28 SBC $2815
2571 4A       LSR
2572 38       SEC
2573 ED 98 28 SBC $2898
2576 A8       TAY
2577 A9 20    LDA #$20
```

```
2579  20  6A  1E  JSR  $1E6A
257C  88          DEY
257D  D0  FA      BNE  $2579
257F  AC  A7  28  LDY  $28A7
2582  4C  56  25  JMP  $2556
2585  C9  45      CMP  #$45
2587  D0  11      BNE  $259A
2589  38          SEC
258A  AD  99  28  LDA  $2899
258D  ED  15  28  SBC  $2815
2590  38          SEC
2591  ED  98  28  SBC  $2898
2594  A8          TAY
2595  A9  20      LDA  #$20
2597  4C  79  25  JMP  $2579
259A  C9  55      CMP  #$55
259C  D0  08      BNE  $25A6
259E  AD  AC  28  LDA  $28AC
25A1  49  01      EOR  #$01
25A3  8D  AC  28  STA  $28AC
25A6  C9  23      CMP  #$23
25A8  D0  1A      BNE  $25C4
25AA  8C  A7  28  STY  $28A7
25AD  AE  9F  28  LDX  $289F
25B0  AD  A0  28  LDA  $28A0
25B3  20  5F  A4  JSR  $A45F
25B6  AC  A7  28  LDY  $28A7
25B9  4C  56  25  JMP  $2556
```

Torniamo ad una routine di stampa: questa sopra viene chiamata appena prima che la riga venga stampata, e controlla vari codici di formato quali la sottolineatura ($2547-$255A predispone per la sottolineatura: dopo ogni carattere la testina della stampante viene fatta tornare indietro per stampare la linea inferiore), la centratura ($2556-$2582), il margine destro ($2585-$2597), ancora la sottolineatura ($259A-$25A3 che torna alla scrittura normale) e la sostituzione del numero della pagina quando viene incontrato il codice di formato '#' ($25A6-$25B9).

```
25C4 AE A8 28 LDX $28A8
25C7 BD EE 28 LDA $28EE,X
25CA 20 6A 1E JSR $1E6A
25CD 4C 56 25 JMP $2556
25D0 AE AB 28 LDX $28AB
25D3 F0 1A    BEQ $25EF
25D5 85 41    STA $41
25D7 29 7F    AND #$7F
25D9 C9 41    CMP #$41
25DB 90 12    BCC $25EF
25DD C9 5B    CMP #$5B
25DF B0 0E    BCS $25EF
25E1 AA       TAX
25E2 A5 41    LDA $41
25E4 29 80    AND #$80
25E6 49 80    EOR #$80
25E8 4A       LSR
25E9 4A       LSR
25EA 85 41    STA $41
25EC 8A       TXA
25ED 05 41    ORA $41
25EF 60       RTS
```

Questa routine si occupa dei codici di formto definiti dall'utente ($25C4-$25CD) e della stampa in ASCII standard ($25D0-$25EF).

```
25F0 20 4E 12 JSR $124E
25F3 38       SEC
25F4 AD 0A 28 LDA $280A
25F7 ED 17 28 SBC $2817
25FA AA       TAX
25FB AD 0B 28 LDA $280B
25FE ED 18 28 SBC $2818
2601 20 5F A4 JSR $A45F
2604 A9 01    LDA #$01
2606 8D 13 28 STA $2813
2609 60       RTS
```

E con la routine di stampa del numero dei byte ancora disponibili per il testo, si conclude la descrizione, certamente sommaria, del funzionamento del programma EASYWORD: un'analisi completa non rientra negli scopi di questo libro Dopo la routine, a partire dalla locazione $2612 trovano posto tutti i messaggi che il programma mostra nelle varie fasi.

1.6 Riassunto dei comandi di Easyword

Vediamo ora due schemi nei quali sono contenuti tutti i comandi e i codici di formato in modo che sia facile la loro consultazione in fase di utilizzo. Eventuali chiarimenti sul funzionamento delle singole azioni è da rivedersi nei paragrafi 1.3 e 1.4.
I comandi sono:

CTRL-A	maiuscolo-minuscolo
CTRL-B	cambia il colore dello schermo
CTRL-D	comando al drive
SH-CTRL-E	elimina il testo dopo il cursore, il *buffer* non viene svuotato
CTRL-E	elimina il testo dopo il cursore, il *buffer* viene svuotato
CTRL-F	definisce un codice di formato
CTRL-G	inserisce 5 spazi
SH-CTRL-H	definisce una parola da ricercare
CTRL-H	ricerca la parola definita da SH-CTRL-H
CTRL-I	modo inserimento-modo normale
SH-CTRL-J	definisce una parola da sostituire
CTRL-J	sostituisce la parola definita da SH-CTRL-J (da usare unitamente a CTRL-H)
SH-CTRL-K	cancella fino a 255 spazi dopo il cursore
CTRL-K	cancella il testo successivo al cursore
CTRL-L	cambia il colore del testo
CTRL-M	equivale alla pressione di 'RETURN'
CTRL-O	stampa il testo
SH-CTRL-O	definisce la periferica per la stampa del testo
CTRL-P	svuota il *buffer*
CTRL-Q	sposta il cursore al periodo successivo
CTRL-R	richiama nel testo il contenuto del *buffer*
CTRL-V	verifica il salvataggio del testo sulla periferica
SH-CTRL-W	elimina il testo prima del cursore, il *buffer* non viene svuotato
CTRL-W	elimina il testo prima del cursore, il *buffer* viene svuotato
CTRL-X	inverte due caratteri
CTRL-Z	va a fine testo
CTRL-£	ricerca e sostituisci automatico
SH-CTRL-2	elimina il testo dopo il cursore, il *buffer* non viene svuotato
CTRL-2	elimina il testo dopo il cursore, il *buffer* viene svuotato

CTRL-4	mostra la *directory* del dischetto
CTRL-7	mostra la memoria-testo ancora libera
CTRL-9	richiama nel testo il contenuto del *buffer*
CTRL-=	definisce un codice di formato
CTRL-;	sposta il cursore di un carattere a destra
SH-RETURN	inserisce 2 'RETURN' e 5 spazi
RUN	inserisce 255 spazi
HOME	cursore in posizione *home* o ad inizio testo
CLR	cancella il testo presente in memoria
DEL	cancella di un carattere prima del cursore
INST	inserisce uno spazio alla posizione del cursore
cur.des.	sposta il cursore di un carattere a destra
cur.sin.	sposta il cursore di un carattere a sinistra
cur. giù	sposta il cursore al periodo successivo
cur.su	sposta il cursore al periodo precedente
F1	sposta il cursore alla parola successiva
F2	sposta il cursore alla frase successiva
F3	sposta il cursore al paragrafo successivo
F4	sposta il cursore alla parola precedente
F5	sposta il cursore alla frase precedente
F6	sposta il cursore al paragrafo precedente
F7	memorizza sulla periferica il testo in memoria
HELP	carica dalla periferica un *file* di testo

I codici di comando sono invece riuniti, assieme ai loro valori di *default* racchiusi tra parentesi, nella seguente lista:

p - lunghezza della pagina (72)
x - larghezza della pagina (80)
l - margine sinistro (5)
r - margine destro (75)
t - margine superiore (5)
b - margine inferiore (66)
h - *header*
f - *footer*
a - stampa in ASCII standard
c - concatenamento file
i - informazioni (fino a 255 caratteri)
j - *linefeed* automatico
m - indentazione sinistra esterna
n - salto pagina
s - spaziatura interlineare (1)

u - sottolinea (va messo prima e dopo)
w - attesa cambio foglio a fine pagina
@ - numero pagina
? - disabilita stampa pagina
- stampa il numero della pagina

CAPITOLO 2

L'ARCHIVIAZIONE ED IL TRATTAMENTO DEI DATI

2.1 Il computer ed i sistemi di archivio

Le due caratteristiche fondamentali di qualsiasi computer che si rispetti sono la disponibilità di memoria e la velocità nell'esecuzione di alcune procedure che, se eseguite con carta e penna, richiederebbero un tempo infinitamente superiore. Accostare queste due prerogative — memoria e facilità nell'operare su quanto è in essa contenuto — significa inevitabilmente pensare alla memorizzazione ed al trattamento dei dati.
Ma quale tipo di dati può essere inserito in un computer, quando esso è un macchina abituata a lavorare in termini di 0 ed 1, mentre le cose che desideriamo archiviare sono in generale complesse come fatture, giacenze di magazzino, oppure i libri della nostra biblioteca? Fortunatamente ci vengono in aiuto due importanti elementi: il programma di archiviazione e gestione dei dati — detto, all'inglese, *DATABASE* — che ci permette sia l'inserimento fisico di questi dati nella memoria del computer che tutte le operazioni su di essi, e l'interprete BASIC contenuto all'interno del C16 che provvede a trasformare i nostri dati in quegli 0 ed 1 comprensibili dal computer.
Poiché, come tutti sanno, spegnendo il computer va perso interamente il contenuto della sua memoria, occorrerà un supporto di *memoria di massa* — generalmente il registratore a cassette o il disk drive — su cui poter immagazzinare in modo permanente i nostri dati al fine di poterli rileggere ed eventualmente modificare in tempi successivi.
Perché, infine, immagazzinare dei dati tramite il computer e non con il solito sistema di archivio composto da carta e penna? Molto semplice: il computer permette di ritrovare un dato tra altri mille in pochi secondi, di ottenere uno stampato di tutto l'archivio o solo dei dati che ci interessano in una particolare situazione, di archiviare l'equivalente di parecchi voluminosi raccoglitori in una sola cassetta magnetica o dischetto da cinque pollici...
I programmi di gestione dati variano in potenza e flessibilità d'uso, a seconda del tipo di periferica di cui si dispone come memoria di massa: in questo libro ne presentiamo due diversi, ed appositamente dedicati rispettivamente all'utilizzo

con il disk drive e con l'unità a cassette. Essi sono totalmente diversi tra loro, in quanto il primo fa pieno uso di tutte quelle caratteristiche offerte dal sistema a dischi — soprattutto velocità di lettura e possibilità di accesso a singoli dati — che sono offerte dall'utilizzo dei cosiddetti *File relativi*, mentre il secondo è appositamente dedicato all'uso dell'unità a cassette con l'utilizzo dei cosiddetti *File sequenziali*.
Occorre a questo punto definire una tripletta di termini — entrati nell'uso comune quando si tratti di gestione dati — che useremo molto spesso nella trattazione che segue. Essi sono:

Record: il record costituisce l'insieme delle informazioni che ci permettono di di definire completamente un certo dato; un esempio di record può essere una fattura, cliente, una giacenza di magazzino.

Campo: il campo è una parte del record o, più intuitivamente, ogni record è composto da più campi. Nel record costituito ad esempio dalla fattura numero 123, i campi potranno essere 7: nome del cliente, suo indirizzo, sua partita IVA, oggetto della fattura, totale imponibile, ammontare dell'IVA, data di emissione della fattura. Nel record di un certo cliente ci potranno ad esempio essere 8 campi: cognome e nome, indirizzo, numero telefonico, dati fiscali, numero ordini effettuati, loro ammontare, suo codice interno, suoi ordini ancora inevasi.

File: il file è l'insieme di tutti i record di cui disponiamo: ci sarà un file per le fatture, uno per i clienti, un altro per i fornitori, uno per le giacenze di magazzino, e così via. Un file può essere ingrandito o ridotto in qualsiasi momento, semplicemente aggiungendovi od eliminandovi dei record.

Passiamo ora ad esaminare più da vicino il primo dei due database, quello dedicato a chi possiede un disk drive 1541 o 1541 compatibile. Anche a chi possedesse soltanto l'unità a cassette consigliamo tuttavia la lettura del paragrafo che segue, in quanto introduce e spiega concetti di grande importanza per un approccio corretto alle problematiche della gestione dei dati computerizzata.

2.2 Le tecniche di programmazione dei database relativi

Un'analogia ricorrente per comprendere appieno l'utilizzo dei programmi database è quella con gli archivi tradizionali. Immaginate di dover catalogare una piccola biblioteca di 500 libri: se non avete il computer, acquisterete un archivio con 500 schede di carta; ogni scheda avrà lo spazio per il nome dell'autore, il titolo del libro, l'anno di pubblicazione, ed il codice da assegnare al libro.
L'intera collezione di schede costitusce dunque il *file*. Ogni scheda, sia essa vuota o compilata, è un *record*. Ogni categoria nella scheda (autore, titolo, anno e codice) è invece un *campo*. Si noti che i record devono avere tutti la stessa

lunghezza, mentre i campi possono averne una qualsiasi (l'autore può essere sia Fo — 2 caratteri — che Pirandello — 10 caratteri — e così via...).
È quindi del tutto immediato vedere come un campo contenga un insieme di caratteri. Naturalmente, ci sarà sulle schede uno spazio limitato per scrivere nome dell'autore, titolo, etc: esisterà dunque un numero massimo di caratteri disponibili per ogni campo, al di sopra del quale si dovrà ricorrere ad abbreviazioni.
Poniamo che la somma dei caratteri disponibili per il nome dell'autore, più quelli per il titolo del libro, più quelli per la data ed il codice sia 72: sarà questa la massima capacità della scheda, e quindi del record.
Dal momento che abbiamo 500 libri e quindi disponiamo di 500 schede, se viene acquistato un nuovo libro, non avremo più schede per catalogarlo. A questo fine acquisteremo una decina di schede nuove, da tenere per ogni evenienza di futuri acquisti. Il contenitore dello schedario ha però una capacità massima — poniamo — di 1200 schede: quando arriveremo a possedere il 1201 esimo libro, dovremo spezzare in due l'archivio.
Poniamo ora che un cliente ci abbia richiesto un libro di Faulkner: se il file (cioè l'archivio) fosse sequenziale — cioè dell'unico tipo usabile con il registratore a cassette — dovremmo consultare le schede in ordine, iniziando dalla prima, dalla A alla Z. Certamente noi umani andremmo 'a naso' ad iniziare la ricerca dove possiamo supporre che ci sia la F, ma il computer non ha 'naso', ed inizierebbe in ogni caso la ricerca dalla lettera A. Pensate al tempo che richiederebbe, in questo modo, la ricerca di un libro di Zoroastro...
Ci sono due buone ragioni per immagazzinare le informazioni nei file relativi: velocità ed economia di memoria. Un file relativo permette infatti un accesso molto veloce ai record individuali.
Con un file relativo, si può andare direttamente nella locazione dove è immagazzinata l'informazione, ed ottenere solo e proprio quell'informazione che stiamo cercando. È un po' come se il computer diventasse dotato di quel 'naso' di cui parlavamo prima, ed anche di un'ulteriore precisione, dal momento che l'uomo inzierebbe a cercare *più o meno* nella posizione in cui ci dovrebbe essere la lettera F, mentre il computer andrà a colpo sicuro ad estrarre la scheda di Faulkner.
Una della ragioni più importanti per l'utilizzo dei file relativi è il fatto che essi non occupano alcuna quantità della memoria del computer, a parte quella occupata da un record individuale.

I programmatori iniziano generalmente a contare da zero: la locazione di memoria più bassa è la 0, e si trova in pagina zero. I file relativi non seguono questa convenzione: il primo record è il numero uno, ed il primo carattere nel record è chiamato carattere numero uno. Inoltre, se avete su disco due o più file relativi, potete averne uno soltanto aperto in un dato momento, anche se è possibile usare contemporaneamente file relativi e sequenziali.
I record, così come sono gestiti da questo programma, possono avere una lunghezza massima di 254 byte.

Il massimo numero di record che è possibile creare è 65535, anche se in pratica non si raggiungerà mai questo limite, sia per le limitazioni di spazio disponibile sul disco, che per le limitazioni di memoria dovute al mantenimento di un indice, come si vedrà tra breve.
Quando un disco viene formattato, la directory mostra 664 blocchi liberi; i blocchi, chiamati anche settori, sono aree del disco che possono contenere 256 caratteri ognuno. Due di essi vengono usati dal DOS (Disk Operating System), lasciando liberi in realtà solo 254 byte per blocco: è questa la ragione per cui la massima lunghezza di un record è 254 caratteri.
I file relativi usando speciali settori, chiamati *side sectors*, per tenere il conto di quali settori contengono dati e quali invece no. Non importa ora conoscere come funziona questo procedimento, basti sapere che ogni file relativo può avere fino a sei 'side sectors'. Ognuno di essi può controllare fino a 120 settori (non si confondano i settori con i record), per un totale quindi di 720 settori, numero superiore a quello dei blocchi liberi in un disco vuoto e formattato.
Riempire completamente un disco con un file relativo, significa usare sei blocchi per i side sectors, lasciando 664-6=658 settori liberi per i dati. Quindi 658 settori di 254 byte ognuno ci lasciano spazio per 167132 caratteri su ogni disco. Questa è la massima capacità di un file relativo su disco.
Ciò significa che si può riempire completamente un disco con 658 record di 254 caratteri ciascuno, oppure 1671 record di 100 caratteri ciascuno, o qualsiasi altra combinazione di numero di record e numero di caratteri per record che, moltiplicati tra di loro, diano come risultato un numero minore od uguale a 167132.
In un file relativo ogni record è numerato, e tutti i record devono essere contraddistinti dalla medesima lunghezza. Normalmente, i campi all'interno di un record hanno anch'essi una lunghezza predeterminata. Per trovare un record all'interno del file, tutto ciò che occorre sapere è il suo numero progressivo. Ciò può suonare come un problema, ma vedremo qualche trucchetto per semplificare il procedimento di ricerca.
Sfortunatamente, la Commodore non ha incluso nel BASIC del C16 alcun comando diretto per la gestione dei file relativi, al contrario invece di quanto accade con il BASIC 7.0 del C128.
Possiamo comunque creare e manipolare i file relativi usando i comandi per la gestione dei file sequenziali, più due comandi per il drive che appariranno meno familiari.
Iniziamo a creare un file relativo. Occorrono tre operazioni:

1. Aprire il file e stabilire la lunghezza dei record.

2. Contrassegnare l'ultimo record del file.

3. Chiudere il file.

OPEN è il comando usato per aprire qualsiasi tipo di file, compreso quindi un file relativo. La forma del comando è solo leggermente diversa da quella usata per

aprire un canale con la stampante, od aprire un file sequenziale su disco:

OPEN1,8,2,"NOME FILE,L,"+CHR$(LUNGH.RECORD)

L'istruzione inzia come qualsiasi altro comando OPEN: 1 è il numero del file logico usato, 8 è il numero di periferica del drive, 2 è il numero del canale di comunicazione usato (può essere indifferentemente un numero tra 2 e 14).
Segue una virgola ed il nome del file tra virgolette, e qui arriva la prima novità: al nome del file, sempre all'interno delle virgolette, segue una virgola, la lettera L, ed un'altra virgola; solo a questo punto si chiudono le virgolette. La lettera L sta per 'Lenght', lunghezza dei record. Questo comando dice al DOS che deve essere aperto un file relativo, la cui lunghezza dei record è contenuta nell'argomento dell'istruzione CHR$. Si ricordi che la massima lunghezza ammessa è 254.
Definire la lunghezza dei record è assolutamente necessario quando si crea un file relativo. Nel momento in cui dovremo eventualmente espandere il file con l'aggiunta di nuovi record, sarà ancora necessario specificarne la lunghezza. Se si lavora con un file relativo già presente sul disco, si può invece usare un semplice comando OPEN:

OPEN1,8,2,"NOME FILE"

Non occorre in questo caso dire al drive che si tratta di un file relativo, esso se ne accorge automaticamente quando lo trova sul disco. Non occorre nemmeno fornire la lunghezza dei record, che allo stesso modo viene automaticamente 'letta' dal drive. Infine, non occorre nemmeno specificarne se il file è stato aperto per la scrittura o per la lettura, in quanto si possono eseguire entrambe le operazioni contemporaneamente.
Il passo seguente aiuta a risparmiare un po' di tempo quando si usa il file: decidiamo il numero di record con il quale — per il momento — dimensionare il nostro file. Potremo aggiungere dei record in qualsiasi momento più avanti. Dal momento che il DOS richiede un certo tempo per creare il file sul disco, conviene aprirlo e creare i record *prima* di immagazzinare qualsiasi dato. Ritornando all'esempio dell'archivio per la biblioteca, è molto più conveniente comprare un certo numero di schede vuote tutte in una volta, piuttosto che perdere tempo per andare in cartoleria ad acquistare una nuova scheda per ogni libro che decidiamo di catalogare.
C'è un altro comando che è utile conoscere: ha questa forma:

```
OPEN15,8,15
PRINT#15,"P"+CHR$(nc+96)+CHR$(bb)+CHR$(ba)+CHR$(pl)
```

Il canale 15 è quello dei comandi per il drive. Dobbiamo in questo caso aprirlo ed inviare al drive sei caratteri. Per primo, la lettera P (per Posizione, o Puntatore). Questa P dice al drive di cercare un certo record. Il secondo carattere è il numero di canale (nc) sommato a 96: se il file era stato aperto con OPEN1,8,2, nc vale 2.

I due numeri seguenti specificano un certo record come somma del suo byte basso più 256 volte il byte alto. Il record numero 300, ad esempio, si tradurrà in CHR$(44)+CHR$(1), dal momento che 44+(1*256)=300. Ricordiamo che il byte basso (bb) ed il byte alto (ba) si calcolano trasformando il numero del record in un numero esadecimale di quattro cifre (300 diventa 012C), spezzandolo in due (01 e 2C), e riconvertendo i due numeri ottenuti in decimale (01 in decimale è 1= byte alto, mentre 2C in decimale è 44= byte basso). Sembra molto complicato, tuttavia non preoccupatevi: queste operazioni non dovrete eseguirle voi, ma sarà il programma a calcolare automaticamente tutti i parametri occorrenti.
L'ultimo carattere (pl), indica in quale parte del record occorre iniziare a leggere o scrivere. In molti casi sarà un CHR$(1) per indicare che vogliamo iniziare dall'inizio del record, al primo carattere. Per iniziare dal sesto carattere, ad esempio, useremo un CHR$(6).
In generale, occorre aprire dapprima il canale 15, quindi aprire il file relativo e posizionare il puntatore. Leggere e scrivere sul file se è necessario, quindi chiudere sia il file relativo che il canale di comando. OPEN15 è la prima operazione da compiere, mentre CLOSE15 sarà l'ultima.
La possibilità di poter modificare un singolo record senza dover riscrivere tutto il file, è una delle caratteristiche che rendono così potente l'uso dei file relativi.
Anche qui si inizia con l'aprire il canale di comando ed il file, posizionare il puntatore sul record, e quindi con il leggere in memoria l'intero record. A questo punto è possibile effettuare tutte le modifiche desiderate, e concatenare tutti i campi all'interno del record come visto precedentemente.
Ci si riposizionerà quindi all'interno del file usando:

```
PRINT#15,"P"+CHR$(nc+96)+CHR$(bb)+CHR$(ba)+CHR$(primo byte)
```

Ora si può usare PRINT# per riscrivere il record. Infine, occorre posizionare il puntatore ancora sul primo carattere del record. Ciò potrà sembrare superfluo, dal momento che ci siamo posizionati sul record prima di scrivere i dati, ma se non si segue correttamente questa procedura, i nostri dati potrebbero essere alterati.
Con i file relativi può essere immagazzinata su disco una gran quantità di dati, tuttavia essi non saranno di alcuna utilità se non siamo in grado di utilizzarli correttamente.
Possiamo caricare in memoria ogni record in qualunque momento ma, con la lentezza del drive 1541, il procedimento può diventare piuttosto lungo e noioso, specialmente se si usano file molto lunghi. Addio ai sogni di grande velocità che hanno fatto nascere l'idea di utilizzare i file relativi...

Esistono fortunatamente delle tecniche che permettono di velocizzare tutto il procedimento. Il modo più comune — che è stato utilizzato nel programma — per gestire una ricerca all'interno di un file relativo, è creare un cosiddetto *file indice* per il campo chiave che si desidera ricercare.
Questo file indice è un sequenziale che viene caricato in un vettore appositamente

creato da un'istruzione DIM. Non è necessario leggere il file indice ogni volta che si operi sul file relativo, ma solo quando si desideri ricercare velocemente un dato record, e non se ne conosca il numero progressivo. Il file indice, come avrete già intuito, associerà il contenuto del campo chiave al numero del record nel file relativo, permettendone una ricerca immediata.

Abbiamo dunque visto che cosa siano i file relativi, e quali siano le istruzioni da usarsi per la loro gestione tramite il BASIC. Analizzeremo ora come si possano creare dei file relativi indicizzati, e passeremo in rassegna alcuni tra i più comuni criteri di ricerca.

Come abbiamo appurato, l'utilizzo dei file relativi permette sostanzialmente due grossi vantaggi: non richiede che l'intero file venga contenuto nella RAM del computer durante le operazioni congiunte di lettura / scrittura, e fornisce l'accesso direttamente al record ricercato, semplicemente specificandone il numero progressivo.

Quest'ultima caratteristica, specialmente se si lavori con database piuttosto capienti, richiede alcune tecniche particolari per essere utilizzata in modo intelligente: raramente o praticamente mai, infatti, quando dovremo ricercare all'interno del file i dati relativi al signor Rossi, ce ne ricorderemo il numero del record. Né del resto potremmo in alcun modo calcolarlo se abbiamo inserito i record in modo casuale e senza un particolare ordine logico.

Anche se procedessimo ad un ordinamento alfabetico dei record in base al cognome, potremmo al massimo arguire che Rossi si troverà piuttosto in fondo al file che al suo inizio, ma dovremo comunque andare per tentativi ricercandone il numero del record. Se le cose stessero veramente così, risulterebbe in fondo ancora più veloce l'utilizzo dei file sequenziali.

Questi ultimi ci vengono invece in aiuto per creare un *indice* del file relativo: proprio come in un libro, dove i vari argomenti trattati possono trovarsi in qualunque sua parte, ricorriamo ad un indice da consultare per poter andare a colpo sicuro nella ricerca di ciò che ci interessa.

Il paragone con il libro è forse molto intuitivo per comprendere bene, ed una volta per tutte, le differenze tra file sequenziali, relativi, e relativi indicizzati: un file sequenziale è come un libro che sappiamo contenga una pagina con dei dati che ci interessano, ma non abbiamo idea di dove questa pagina si trovi al suo interno; dovremo necessariamente iniziare a sfogliare il libro a partire dalla prima pagina, fino a che ci capiterà sotto gli occhi ciò che cerchiamo.Un file relativo ci fornisce invece l'accesso ad una singola pagina,ma dobbiamo ricordarcene il numero per poter andare a colpo sicuro per ritrovare ciò che ci interessa.

Infine, un file relativo indicizzato è come un normale libro: per trovare il numero di pagina cercato, andiamo a colpo sicuro a leggere l'indice del libro che ci fornirà, con una breve ricerca *sequenziale* al suo interno dalla A alla Z, l'argomento di interesse ed il suo numero di pagina.

I dati contenuti in un record sono suddivisi in vari campi — ed esempio cognome, nome, via, città, etc. Dovremo stabilire quale sia il campo più importante per l'identificazione di un certo record, che ci permetterà di eseguire la ricerca. In un

database di indirizzi è evidente che questo campo *chiave* sia il cognome, mentre per un database di una biblioteca può essere il codice del libro, od il suo titolo. I campi chiave — così detti perché ci forniscono una chiave di accesso al record — all'interno di un record possono anche essere più d'uno e, per non complicare ulteriormente le cose e per attenerci alla struttura del programma, assumeremo di trattare un solo campo chiave.
L'indice di un libro ha la funzione di associare ad un certo argomento il numero di pagina relativo alla posizione di quell'argomento nel libro; il nostro indice avrà quindi la funzione di associare ad un certo contenuto del campo chiave il numero del record al quale quest'ultimo appartiene.
Con la piccola spesa di un po' di spazio sul disco ed in memoria, si può creare un indice del tipo seguente:

FILE INDICE		FILE RELATIVO			
CHIAVE	N. RECORD	CAMPO1 (CHIAVE)	CAMPO2	CAMPO3	CAMPO4
Bianchi	3	Bianchi	Mario	Diaz,3	Milano
Grigi	1	Grigi	Giovanni	Orefici,2	Milano
Neri	4	Neri	Sergio	Moreri,14	Trieste
Verdi	2	Verdi	Antonio	Francia,65	Torino
...	...	...	...	...	...

Il file indice sarà molto più piccolo in dimensioni rispetto al file relativo cui si riferisce, essendo formato semplicemente da un elenco dei contenuti di un solo campo, associati al relativo numero di record. Il file indice è un sequenziale: esso può essere caricato in RAM una volta per tutte all'inizio del programma con i suoi contenuti inseriti in un apposito vettore, oppure può essere lasciato sul disco, e letto con delle INPUT# fino a che non si ottenga l'informazione desiderata.
Poiché, se si utilizzano i file relativi, raramente si pongono problemi di occupazione eccessiva di memoria RAM, risulta assai più conveniente il primo metodo, in quanto esso offre, con l'unico scotto di un minimo di occupazione di memoria, i seguenti vantaggi:
— estrema velocità di ricerca di un campo chiave, essendo tutti i dati organizzati all'interno di un vettore nella RAM del computer.
— possibilità di veloci aggiunte all'indice quando si inseriscano nuovi record, o cancellazioni nel caso opposto.

È inoltre sufficiente un vettore monodimensionale per contenere i campi chiave: il numero del record ad essi associato potrà essere fatto coincidere con la posizione della data stringa all'interno del vettore stesso.
Un altro sistema usato da programmi commerciali e professionali, è la cosiddetta codifica 'hashing': vediamo di che cosa si tratta.
Non esiste più un indice separato dei campi chiave con i numeri di record associati, ma uno speciale algoritmo provvede ad associare ad ogni campo

chiave — direttamente all'interno del file relativo — un *codice hash*, calcolato in base alla combinazione dei vari caratteri che costituiscono il campo chiave stesso. Praticamente si ottiene, per vie traverse, un accesso diretto al record non più in base al suo numero, ma in base al contenuto del suo campo chiave (in realtà l'accesso fisico al disco avviene sempre in base al numero: è l'accesso *da programma* che avviene in base alla sequenza di caratteri del campo chiave).
I campi chiave possono essere formati da lettere, numeri, o da un miscuglio di lettere e numeri; quasi sempre sono totalmente alfabetici (ad esempio nel caso del citato archivio indirizzi) ma, nel caso per esempio di una biblioteca, possono essere alfanumerici come la maggior parte dei codici assegnati ai libri.
Se si pensa al numero di possibili combinazioni tra i vari caratteri che possono costituire il campo chiave, si avrà un'idea della varietà di possibili situazioni che si possono presentare: consideriamo ad esempio un campo chiave totalmente alfabetico e contenente dei cognomi, e poniamo che esso sia costituito al massimo da dieci caratteri.

Dal momento che ci sono 21 lettere nell'alfabeto, esistono 21 elevato alla 10 modi di combinare questi caratteri nel nostro campo chiave. 21 alla 10 è un numero vicino a 10 alla 11: un numero enorme, anche se molte delle possibili combinazioni non si verificheranno mai all'atto pratico — non troveremo mai dei cognomi come ZZZZZZZZZZ o come QTSBCDZFHL.

Diciamo comunque che almeno 10 elevato alla 6 combinazioni potranno dare come risultato dei cognomi razionalmente possibili; rimangono quindi circa un milione di possibili permutazioni. Questi dati ci portano alla conclusione che, per gestire un sistema d'accesso ai record in base al campo chiave e senza ricorrere a particolari trucchi, dovremmo tenere in memoria un milione di possibili chiavi e di volta in volta confrontarle con il cognome che ci serve: probabilmente non ci riusciremmo nemmeno disponendo di un mainframe.

Il sistema di hashing consiste invece nel mappare ogni campo chiave solo all'interno dell'intervallo numerico costituito dai record allocati nel file.
Esistono principalmente due diversi sistemi di approccio al problema: il sistema deterministico e quello cosiddetto di 'randomizzazione'. Il primo conta su un algoritmo basato sulla conoscenza a priori di un certo numero di campi chiave, e della loro distribuzione all'interno del file: ogni modifica su quest'ultimo, come la semplice aggiunta di un record, richiede tuttavia la riscrittura dell'intero algoritmo.
Il secondo metodo si divide a sua volta in due diversi tipi di sottoprocedure: quelle basate su algoritmi che cercano, il più possibile, di mantenere inalterato l'ordine originario dei campi chiave, e quelle con algoritmi che si basano sull'unicità dei campi chiave. L'algoritmo di hashing è uno di questi ultimi.

Passiamo ad un esempio pratico per vedere come funziona un algoritmo di hashing: si abbia un file relativo composto da 50 record, il cui campo chiave sia a sua volta costituito da un insieme di 10 caratteri. Poiché l'algoritmo si proprone di calcolare il numero del record in base al dato contenuto del campo chiave, l'enorme numero possibile di record visto precedentemente (più di un milione) viene dapprima ridotto al numero effettivo dei record da gestire. Il campo chiave

sarà quindi elaborato dall'algoritmo, per cercare di risalire all'effettivo numero del record.
Un algoritmo di hashing prende un numero e lo scompone in maniera alquanto disorganizzata: più il numero che ne risulta può essere considerato casuale, più è da considerarsi sofisticato l'algoritmo stesso. Oltre a 'rimescolare' il numero, esso viene ridotto ad un numero di cifre inferiore; un esempio di un semplice hashing consiste nell'elevare a potenza 2 un numero, e ricavarne le due cifre centrali:

numero: N=8192 — N ↑ 2=67108864 —> H= 08
numero: N=9234 — N ↑ 2=85266756 —> H= 66

oppure dividere il numero N (cioè il campo chiave) per il numero dei record, ed assumere come risultato il resto della divisione: sia N=8192 con 10 record; il risultato dell'hashing sarà 92.
Se il campo chiave è composto da caratteri alfabetici, si applicano le stesse tecniche convertendo la stringa in un unico numero. La semplice somma dei codici ASCII dei vari caratteri costituenti la stringa, non offre sufficienti garanzie di univocità nella determinazione (uno stesso numero potrebbe corrispondere a diverse stringhe, tipo TRENO e TERNO).
Per ottenere una maggiore univocità, si possono ad esempio sommare in modo cumulativo i prodotti tra i codici ASCII costituenti coppie di caratteri, partendo da un lato della stringa, oppure utilizzare altri sistemi matematicamente più rigorosi, che non staremo tuttavia ad elencare.
Ovviamente la tecnica di hashing è inapplicabile per la ricerca dei record in un file relativo generico: il file deve invece — ovviamente — essere scritto in modo che il numero di ogni suo record coincida con l'hashing del suo campo chiave.
La metodologia dell'hashing deve essere dunque impiegata già in fase di scrittura del file, altrimenti non sussisterebbe alcuna relazione logica tra contenuto del campo chiave e numero del record.
La ricerca vera e propria avverrà partendo dal record il cui numero è dato dal risultato dell'hashing e, in caso il campo chiave di quel record non sia coincidente con quello 'sorgente' dell'hashing, la ricerca procederà all'interno del file.
Come si vede, i sistemi per aggirare l'ostacolo costituito dalla mancata conoscenza del numero di un dato record sono molteplici. Essi, pur implicando una ulteriore complessità nel programma database, permettono di raggiungere risultati notevoli, con una velocità di accesso al record di tutto rispetto anche per drive lenti come il 1541.

2.3 Relfiling system 1.0: un database relativo per sistema a dischi

Passiamo ora al programma vero e proprio, esaminando le sue caratteristiche interne ed il modo di utilizzarle.

La quasi totalità di programmi database esistenti per il C16 nasce per l'utilizzo con il registratore a cassette, e l'opzione per l'uso del disk drive, pur quasi sempre presente, altro non fa che dirigere l'output su questa periferica invece che sul registratore. Il modo con il quale vengono gestiti i dati rimane quello proprio dell'unità a cassette, e rappresenta un vero e proprio spreco di tempo e velocità per l'utilizzatore del drive.

Le possibilità offerte dal drive sono invece molto più ampie e, se il programma di gestione dati è stato scritto per sfruttarle, si può ottenere un risparmio del tutto ragguardevole nel tempo di attesa per l'accesso ai dati.

I file relativi, disponibili solo con l'utilizzo del drive ed introdotti nel paragrafo precedente, rappresentano uno strumento molto potente per l'archiviazione di dati su disco.

Non più un accesso sequenziale ai dati: non occorre più leggere cinquecento record che non interessano per caricare il cinquecentunesimo, non più pesantissime limitazioni sul numero dei record e la loro lunghezza a causa del rapido decrescere della memoria disponibile...

Con questo programma potrete caricare in memoria il record che vi serve in meno di un secondo, sia esso il primo record del file piuttosto che il centesimo. Potrete modificare o riscrivere un qualunque record senza dover risalvare tutto il file dati, e potrete disporre di un'opzione di ricerca veloce dei record che, lavorando in RAM, è dotata di tempi di attesa bassissimi.

Una volta caricato il programma dalla cassetta allegata al volume, salvatelo su disco con un nome di vostro gradimento. Al RUN vi verrà domandato se intendete utilizzare la stampante: tenete conto che, in caso di risposta affermativa, la maggior parte dell'output che normalmente avviene su video verrà indirizzato alla stampante.

Per i primi esperimenti con il programma, decidiamo di non usare la stampante. Premete dunque N (o soltanto RETURN, dal momento che la risposta negativa è di default) alla prima richiesta del computer.

Apparirà il menù principale, composto dalle seguenti opzioni:

1. Creazione nuovo file
2. Definizione file corrente
3. Lettura file corrente
4. Scrittura file corrente
5. Ricerca di un record
6. Opzioni di stampa
7. Memoria libera
9. Fine operazioni

Dal momento che non possediamo un file già pronto, dovremo crearcelo. Selezioniamo qiundi l'opzione 1.

La cosa migliore, quando si utilizzino dei file dati su disco, è quella di dedicare un apposito disco al file dati, mentre il programma database può essere registrato

su un qualunque altro disco insieme ad altri programmi di utilità.
I file relativi possono infatti avere dimensioni del tutto ragguardevoli, ed occupare quindi una buona parte dello spazio disponibile sul disco.
Alla domanda del computer "Inserisci disco dati e premi RETURN, oppure premi spazio per generarlo", sceglieremo di generare il disco dati. Premete dunque la bassa spaziatrice ed inserite un disco vergine nel drive; dopo alcune conferme di procedura, il computer vi chiederà il nome del database — che assegnerà come nome al disco — ed il codice identificatore del disco stesso, composto come sempre da due caratteri qualsiasi.

Dopo la formattazione, verrà visualizzato il numero dei blocchi liberi per il file dati. Come noterete, questo numero non è 664: sei blocchi vengono infatti usati per i "side sectors"' del file relativo. Il disco è ora pronto, non resta che definire come vogliamo il nostro file di dati.
Ci serve una rubrica telefonica, un archivio dischi, un archivio fatture o un archivio di magazzino? Nessun problema, il programma ci crea un file 'su misura' per le nostre esigenze.
Prima di procedere con gli input del programma, conviene munirsi di un pezzo di carta ed una matita, e stabilire che tipo di dati desideriamo siano trattati dal calcolatore.

Un record è diviso in campi: il classico esempio della rubrica telefonica vede un record tipicamente diviso in cinque campi: cognome e nome, via, città, numero telefonico, note varie. Un archivio fatture di un professionista potrebbe ad esempio essere formato da record con otto campi: ragione sociale, indirizzo, partita IVA, codice fiscale, data, motivo, ammontare dell'IVA, totale fattura.
Stabilite quindi che cosa desiderate archiviare, e scrivete sul foglio di carta i nomi dei campi che vi servono. Una volta definito il numero di campi ed il loro nome, dovremo decidere da quanti caratteri deve essere composto ciascun campo: per il codice fiscale ci vorranno ad esempio 16 caratteri (un numero minore sarebbe insufficiente, ed un numero superiore sarebbe uno spreco di spazio sul disco), per un cognome potrebbero bastare 15 caratteri, e così via.
Una volta stabilito esattamente quale aspetto dovrà avere il nostro database, si potrà rispondere alle domande del computer sul numero dei campi, il loro nome e la loro lunghezza. Si noti che il primo campo viene assunto dal programma come *campo chiave* (campo sul quale verranno effettuate le ricerche all'interno del database) e dovrà quindi contenere l'informazione più importante del record.
La lunghezza del primo campo è limitata ad un massimo di 79 caratteri, mentre gli altri campi possono avere una lunghezza anche superiore. Si tenga tuttavia presente che un record non può essere più lungo complessivamente di 254 caratteri, compreso l'ultimo carattere di fine record automaticamente aggiunto dal programma.
La somma delle lunghezze dei vari campi non potrà quindi superare il totale di 253 caratteri, mentre il numero dei campi non è soggetto a vincoli.
Il programma procederà quindi domandando le dimensioni del file, cioè il numero totale di record. È conveniente assegnarvi un valore il più alto possibile, in quanto

non sarà più possibile, raggiunto il numero di record pari alla capacità assegnata, aggiungere nuovi record al file. Il massimo numero di record consentito dal programma non è fisso, ma dipende da due diversi fattori che interagiscono tra di loro: spazio disponibile sul disco per ospitare il file dati, e spazio disponibile in memoria per ospitare il file indice.
Nel C16 inespanso, il secondo è proporzionalmente molto inferiore al primo, e quindi il massimo numero di record è limitato dalla memoria disponibile. Poiché quest'ultima è influenzata anche dal numero di campi che abbiamo precedentemente definito per ogni record, non esistono criteri ben definiti per stabilire un tetto superiore al numero di record in un file. La cosa migliore è procedere per tentativi decrescenti, partendo da un numero di record di circa 200 per ogni file: se il programma non segnala — attraverso la sua propria routine di intercettazione — alcun errore di 'OUT OF MEMORY' il numero è accettabile, altrimenti occorrerà ridurlo fino a che il programma non segnali più alcun errore.
Se si collega invece al C16 un'espansione di memoria (quasi sempre le espansioni di memoria per il C16 sono in grado di portarlo a 64 Kbyte) non ci sarà alcun problema per le dimensioni dei file, che potranno agevolmente superare i 1000 record.
Stabilite le massime dimensioni del file, dovremo scegliere un nome da assegnargli: chiamandolo "PROVA".
Dopo un tempo di attesa necessario per creare sul disco il file indice (un sequenziale che contiene tutti i parametri più significativi del file dati) e per creare lo spazio per il file dati stesso, si ritornerà al menu principale.
Utilizziamo ora l'opzione 2 per comunicare al programma il file su cui intendiamo lavorare, e rispondiamo con "PROVA" alla sua domanda.
A questo punto potremo iniziare ad inserire informazioni attraverso l'opzione 4: comparirà una linea in campo inverso nella parte alta dello schermo contenente le opzioni usabili in modo inserimento dati.
Esse sono:

tasto "-" passa al record precedente
tasto "+" passa al record successivo
tasto "£" passa ad un record particolare
tasto "E" ritorna al menu principale
tasto "I" permette l'inserimento dati.

A questo punto saremo posizionati sul record 1. Premiamo il tasto I per l'inserimento dati. Comparirà il nome del primo campo con tra parentesi la sua lunghezza, e saremo pronti per scrivere i nostri dati. Se, scrivendo, raggiungiamo la fine del campo, il computer passerà automaticamente al campo successivo; se invece i dati che desideriamo inserire hanno una lunghezza inferiore alla massima lunghezza del campo, dovremo premere RETURN per passare al campo successivo. Se non si desidera scrivere su un campo, è sufficiente premere RETURN. Dopo aver inserito i dati in tutti i campi, il record verrà scritto sul disco, ed il programma si posizionerà sul record successivo, in questo caso il numero 2.

Un record può essere riscritto in qualsiasi momento, posizionandosi su di esso con i tasti "+" e "-" oppure con il tasto '£' (lira) che permette di scrivere su un qualunque record comunicandone il numero al computer. Premendo quindi il tasto "I" si potrà riscrivere il record.
Scriviamo anche i dati del record 2 e ritorniamo al menu principale premendo il tasto "E".
Il nostro file conterrà ora due record allocati, che potremo leggere con l'opzione 3. La schermata che compare è analoga a quella di scrittura, ed identiche sono le funzioni dei tasti (a parte il tasto "I" che ovviamente non compare in questa opzione). Il programma si posizionerà automaticamente sul primo record e lo visualizzerà sullo schermo. Potremo leggere il secondo sia premendo "+" che premendo "£" e rispondendo con 2 alla domanda del computer. Dal momento che abbiamo inserito soltanto due record, cercando di leggere un record non scritto, ad esempio il numero 3, il computer ci avviserà che tale record è vuoto.
Premendo "E" ritorniamo al menu principale. Esaminiamo ora le opzioni di stampa, selezionando l'opzione 6.
Il menu di stampa è composto dalle seguenti opzioni.

1. Stampa indice contenuto
2. Stampa singoli record
3. Stampa l'intero file
4. Stampa particolari campi
5. Ritorna al menu principale

Poiché inizialmente abbiamo deciso di non utilizzare la stampante, tutta la stampa avverrà molto velocemente su video, e potrà essere rallentata tenendo premuto il tasto Commodore.
L'opzione 1 provoca la stampa dell'indice. L'indice è basato sul campo chiave che, come detto, è il primo campo di ogni record: verrà stampato il numero progressivo dei record con a fianco il rispettivo contenuto del campo chiave.
L'opzione 2 funziona in modo assolutamente analogo all'opzione di lettura dei record vista precedentemente, provocandone però la stampa; il tasto '£' (lira) può essere utilizzato per posizionarsi su un qualunque record.
L'opzione 3 stampa l'intero file, record per record e campo per campo.
L'opzione 4 permette infine la stampa solo di un particolare campo, che verrà comunicato al programma attraverso il suo numero (campo 1, campo 2, campo 3, etc.); la stampa avviene per tutti i record allocati nel file.
Ritorniamo al menu principale selezionando l'opzione 5.
Una delle caratteristiche più utili dei programmi di archivio, è la possibilità di ricercare un certo record all'interno di un file che magari ne contiene più di mille.
L'opzione 5 esegue appunto questa funzione.
Il computer ci chiederà il contenuto del campo chiave che, ricordiamo, è il primo campo. Volendo, ad esempio, ricercare il record contenente i dati del signor Rossi, digiteremo "Rossi". Il programma ricercherà quindi, all'interno dell'indice del file, tutte le occorrenze del nome "Rossi". Se questo nome compare una sola

volta (c'è un solo Rossi nell'archivio), verrà visualizzato il record contenente i suoi dati. Se, al contrario, esistono più record il cui campo chiave contiene il cognome "Rossi", il programma avvertirà che esistono n record il cui campo chiave è "Rossi": si potrà quindi premere "C" per visualizzarli uno per uno, oppure premere "E" per ritornare al menu.

Una particolarità interessante dell'opzione di ricerca, è che essa permette di ricercare un particolare record semplicemente digitando solo una parte del suo campo chiave. Per spiegarci meglio, se il file contiene un record per il signor Rossi, uno per il signor Rossetti, ed uno per il signor Rosellini, ricercando "Rossi" verrà trovato solo il record del signor Rossi. Se, invece, alla domanda del computer sul contenuto del campo chiave digitiamo "Ross", il programma troverà sia "Rossi" che "Rossetti". Se digitiamo "Ros" verrà invece trovato anche "Rosellini". Se digitiamo "R" verranno trovati tutti i record che iniziano per R.
Questa particolarità della funzione di ricerca risulta estremamente utile se non ricordiamo esattamente un cognome, o la forma del dato che ci serve.
Ritorniamo al menu principale. L'opzione 7 ci dice l'ammontare della memoria disponibile (dopo un tempo di attesa più o meno lungo necessario al computer per effettuare i suoi controlli). Dal momento che l'indice del file viene mantenuto in RAM al fine di sveltire le operazioni su di esso quali la ricerca di un nome, vengono dimensionati dei vettori per contenere quei dati importanti quali il nome dei campi, la loro lunghezza, il contenuto del primo campo, etc. Se il file dati è molto capiente, è opportuno prima di ogni ampliamento controllare l'ammontare di memoria libera di cui si dispone, per evitare uno spiacevole messaggio di "OUT OF MEMORY".

Il programma è comunque dotato di un controllo interno, che segnala condizioni di errore generate dalla mancanza di memoria libera, nel caso il file si estenda in misura troppo elevata per essere gestita dalla memoria del C16.
L'opzione 8 è la più importante in assoluto.
Quando abbiamo finito di lavorare su un certo file, dobbiamo *sempre* usare l'opzione 8. Essa infatti provvede a scrivere sul disco l'indice aggiornato del file, il suo numero di record allocati, ed altri importantissimi parametri che, se mancanti, non ci permetterebbero più l'accesso al file dati.

È opportuno usare questa opzione anche se si è aperto il file per sola lettura dei dati e non si è operata alcuna modifica. L'uso congiunto delle opzioni 8 e 2 permette il rapidissimo accesso a file differenti, e verrà discusso tra poco.
Premiamo dunque il tasto 8: il file indice aggiornato con i due record che abbiamo precedentemente inserito verrà scritto sul disco, ed il programma domanderà se si tratta della fine dei lavori oppure no. Rispondiamo negativamente: il programma viene reinizializzato.
Decidiamo ancora di non utilizzare la stampante; anche questa volta, all'apparizione del menu principale, dovremo comunicare al programma con quale file dati abbiamo intenzione di lavorare. Selezioniamo quindi l'opzione 2: il programma scandaglierà il disco alla ricerca dei file dati presenti (tralascerà invece qualunque altro file che non sia una file dati). Comunichiamogli che intendiamo lavorare sul

file “PROVA”. Se avessimo sullo stesso oppure su un altro dischetto dei diversi file dati, con questa opzione potremmo accedere indifferentemente ad uno qualunque di questi file.

Il computer leggerà il rispettivo file indice e ci comunicherà la composizione del file: suo nome, numero di record, numero di campi, loro nome e rispettiva lunghezza, ed infine il numero dei record allocati. Per il file “PROVA”, il numero di record allocati sarà 2. Abbiamo a questo punto ‘regolato’ il programma sul file “PROVA”, e possiamo aggiungervi dei dati, leggerne altri, o compiere qualsiasi altra operazione nel modo analizzato precendentemente.

Ricordiamo tuttavia di uscire *sempre* da questa ‘regolazione’ del programma attraverso l’opzione 8 del menu principale.

L’utilizzo del programma è molto semplice ed intuitivo, e permette con l’uso congiunto delle opzioni 2 ed 8 di passare rapidamente da un file all’altro. Con un minimo di sperimentazione e di pratica si diverrà in grado di operare molto velocemente su diversi file, leggendo e scrivendo i dati con una affidabilità e velocità sconosciuta ai precedenti programmi di gestione dati per C16.

I file relativi posseggono — come abbiamo visto — il grande vantaggio di rendere possibile l’accesso diretto ad un qualunque record, semplicemente specificandone il numero distintivo.

Tuttavia, nella maggior parte dei casi, l’utilizzo pratico di un archivio richiede che un certo record possa essere ricercato in base al suo contenuto di un certo campo, e non in base al suo numero progressivo all’interno del file, in quanto quest’ultimo è del tutto sconosciuto all’utente.

Il problema si risolve creando un indice del file che raccoglie sequenzialmente i contenuti di tutti i campi chiave dei vari record. Il vettore CC$, dimensionato sul numero massimo di record, raccoglie appunto il contenuto del primo campo di tutti i record del file. La posizione di un certo dato all’interno del vettore coincide con il numero del record relativo. Se, ad esempio, CC$(3) è uguale a “Rossi”, il programma saprà che il record contenente i dati del signor Rossi è il numero 3, e così via. Questo sistema di indicizzazione è quello utilizzato dal database qui presentato, ed è caratterizzato da un pro ed un contro. Il pro è che si riesce a gestire l’indice con un solo vettore dimensionato sul numero massimo di record, limitando l’occupazione di memoria ed aumentando di conseguenza il massimo numero di record gestibile dal programma. Il contro è che questo sistema non permette ordinamenti alfabetici dell’indice, in quanto verrebbe sconvolto il legame tra contenuto del campo chiave e numero del record: facendo un sort all’interno di CC$ nell’esempio precedente, probabilmente “Rossi” andrebbe a capitare in posizione — poniamo — 128, mentre il suo record rimarrebbe comunque il numero 3. Per rendere possibile l’ordinamento bisognerebbe prevedere una matrice bidimensionale invece di un vettore, CC$(n,n), dove la prima colonna sia composta dal contenuto dei campi chiave, e la seconda dal relativo numero del record. L’opzione di ordinamento si paga quindi in termini di maggior occupazione di memoria e conseguentemente minor numero di record gestibili dal database. Un’altra curiosità del programma è l’immagazzinamento in memoria del numero

di canale della stampante con istruzioni POKE e non con delle semplici variabili: a causa della possibilità di passare agevolmente da un file dati all'altro offerta dal programma, sono necessarie molte istruzioni CLR per azzerare i dimensionamenti dei vettori. Poiché un'istruzione CLR azzera anche il contenuto di tutte le variabili e chiude tutti i canali di comunicazione con le periferiche, se il numero di canale della stampante è invece contenuto in una locazione di memoria non risente di questo generale azzeramento.
Ecco ora un elenco delle principali variabili e vettori utilizzati nel programma:

RN	massimo numero di record
TY	numero di record allocati
F$	nome del file dati
RL	lunghezza dei record
C	numero dei campi
N$ ()	nome dei campi
L ()	lunghezza dei campi
CC$ ()	contenuto dei campi chiave
RC$	contenuto del record
PR	canale stampante
R$	ritorno carrello
PO	posizione di lettura record
HB	byte alto puntatore al record
LB	byte basso puntatore al record
E1$	numero codice di errore del drive
E2$	messaggio di errore del drive
E3$	messaggio di errore maiuscolo
BY	ammontare byte liberi
LS	lunghezza di una stringa
CT ()	vettore lavoro per ricerca record
F	variabile di flag
A	codice ASCII di un tasto

2.4 Commento al listato

100-110 inizializzazione dei colori di fondo nero e testo bianco e passaggio in modo maiuscolo/ minuscolo

120-130 subroutine di visualizzazione dell'ammontare di memoria libera.

150-180 scelta se utilizzare o meno la stampante ed apertura del canale di comando del drive.

200-290 submenu per le opzioni di stampa.

310-430 subroutine di stampa dell'intero contenuto del file dati.

440-450 stampa del singolo record: viene utilizzata la normale routine di visualizzare malgrado l'output avvenga su stampante (la variabile PR vale 4, come il canale della stampante).

480-570 stampa di particolari campi di tutti i record del file. Un particolare campo si ottiene posizionando il puntatore al record sul carattere di inizio campo, e limitando quindi la lettura del disco al numero di caratteri contenuti nel campo stesso.

590-770 ricerca di un record all'interno del file e controllo sull'esistenza eventuale di più record dotati dello stesso contenuto del campo chiave. La ricerca avviene con il principio della 'wild card'.

840-1000 subroutine per l'inserimento dei dati all'interno dei record. Si ricorre ad un input controllato con generazione di uno pseudo-cursore tramite delle POKE in mappa video ed in mappa colore. Se la lunghezza della stringa digitata è inferiore alla massima lunghezza del campo relativo, vengono aggiunti degli spazi shiftati (CHR$(160)) fino a fine campo.

1010-1120 subroutine di lettura del file indice con visualizzazione ed inizializzazione del programma su massimo numero di record, numero di record allocati, nome e numero dei campi, etc.

1140-1150 subroutine che attende la pressione di un qualunque tasto, utilizzata più volte dal programma. La subroutine restituisce anche il codice ASCII del tasto premuto.

1170-1200 subroutine di lettura del canale degli errori del drive: il messaggio ottenuto viene convertito in lettere maiuscole all'interno della variabile E3$.

1210-1220 subroutine di posizionamento del puntatore al record in funzione del valore assunto dalle variabili LB, HB e PO.

1240-1370 menu principale di selezione del programma.

1380-1400 definizione del file corrente sul quale si desidera operare.

1410-1420 subroutine di ciclo di attesa di qulche secondo.

1440-1620 inizializzazione generale del programma con istruzioni di lettura del disco dati o di una sua generazione.

1640-1700 subroutine di calcolo per il numero di blocchi disponibili per il file dati, all'interno del disco.

1720-1870 definizione di tutti i parametri fondamentali di un nuovo file all'atto della sua creazione sul disco.

1890-2060 creazione effettiva sul disco del file dati (allocandone i blocchi necessari alla sua gestione).

2080-2220 subroutine di inserimento dati all'interno del record, con possibilità di scrittura su un record qualunque.

2250-2310 scrittura effettiva del record all'interno del file, nella posizione desiderata.

2330-2520 subroutine di lettura di un qualunque record all'interno del file.

2540-2630 riscrittura del file indice con i parametri aggiornati relativi al file dati. Il programma domanda quindi se si intende continuare nei lavori: in caso negativo viene effettuato un parziale reset del computer.

2650-2710 subroutine di intercettazione degli errori, particolarmente utile in fase di definizione del file, per evitare blocchi del programma a causa del superamento della RAM disponibile.

2730-2880 subroutine di analisi del disco: viene esaminata la directory alla ricerca di file dati generati da questo programma, e vengono visualizzati i nomi ad essi assegnati.

RELFILING SYSTEM 1.0

```
100 COLOR0,1:COLOR4,1:COLOR1,2:SCNCLR
110 CLR:PRINTCHR$(14):GOTO150
120 PRINT"{CLR}{SH A}TTENDERE...":BY=FRE
    (0)-(FRE(0)<0)*65536
130 PRINT"{CLR}"BY"BYTE LIBERI":GOSUB142
    0:GOTO1240
140 REM **LA LOCAZIONE 819 CONTIENE IL #
     DI CANALE PER LA STAMPANTE**
150 INPUT"{CLR}{SH U}TILIZZO DELLA STAMP
    ANTE (S/N){2 SPC}N{3 CUR.SIN}";A$
160 IFA$<>"S"THENPOKE819,2:GOTO180
170 IFA$="S"THENOPEN4,4,7:POKE819,4:GOTO
    180
```

```
180 OPEN15,8,15:OPEN2,3:GOTO1240
190 REM **OPZIONI DI STAMPA**
200 PRINT"{CLR}{RVS ON}{12 SPC}{SH O}{SH
     P}{SH Z}{SH I}{SH O}{SH N}{SH I} {S
    H D}{SH I} {SH S}{SH T}{SH A}{SH M}{
    SH P}{SH A}{11 SPC}"
210 A$="{5 CUR.DES}":B$="{9 CUR.GIU}":PR
    =PEEK(819)
220 PRINTB$;A$;"1. {SH S}TAMPA INDICE CO
    NTENUTO"
230 PRINTA$;"{CUR.GIU}2. {SH S}TAMPA SIN
    GOLI RECORD"
240 PRINTA$;"{CUR.GIU}3. {SH S}TAMPA L'I
    NTERO FILE"
250 PRINTA$;"{CUR.GIU}4. {SH S}TAMPA PAR
    TICOLARI CAMPI"
260 PRINTA$;"{CUR.GIU}5. {SH R}ITORNA AL
     MENU PRINCIPALE"
270 GOSUB1140:IFA<49ORA>53THEN270
280 ONA-48GOSUB310,450,370,470
290 GOTO1240
300 REM **STAMPA INDICE**
310 IFCC$(1)<>""ANDCC$(2)<>""ANDCC$(3)<>
    ""THEN330
320 PRINT"{CLR}{SH I}NDICE NON IN MEMORI
    A!":GOSUB 1420:GOTO200
330 PRINT#PR,"{SH F}{SH I}{SH L}{SH E}:"
    F$TAB(40)"{SH R}ECORD ALLOCATI:"TY:P
    RINT#PR:PRINT#PR
340 FORI=1TOTY:PRINT#PR,"{SH R}ECORD {SH
     N}.";I;TAB(40)"{SH C}HIAVE: "CC$(I)
    :NEXT
350 PRINT#PR:RETURN
360 REM **STAMPA L'INTERO FILE**
370 OPEN1,8,2,"0:"+F$:PRINT#PR,"{SH F}{S
    H I}{SH L}{SH E}:"F$TAB(40)"{SH R}EC
    ORD:"TY:PRINT#PR:PRINT#PR
380 FORI=1TOTY:RC$="":PRINT#PR:PRINT#PR,
    "{SH R}ECORD {SH N}.";I;":":PRINT#PR
```

```
390 HB=INT(I/256):LB=I-HB*256
400 PO=1:GOSUB1220:PS=1:FORK=1TORL
410 GET#1,A$:RC$=RC$+A$:NEXT
420 FORK=1TOC:PRINT#PR,N$(K)": "MID$(RC$
    ,PS,L(K)):PS=PS+L(K):NEXT
430 NEXTI:CLOSE1:PRINT#PR:RETURN
440 REM **STAMPA RECORD SINGOLO**
450 GOTO2330
460 REM **STAMPA PARTICOLARI CAMPI**
480 PRINT"{CLR}";:FORI=1TOC:PRINTI,N$(I)
    :NEXT
490 INPUT"{CUR.GIU}{SH N}UMERO DEL CAMPO
     DA STAMPARE";A:IFA>CORA=0THEN490
500 L=1:FORK=2TOA+1:L=L+L(K-2):NEXT
510 OPEN1,8,2,"0:"+F$:FORI=1TOTY
520 HB=INT(I/256):LB=I-HB*256:PO=L
530 GOSUB1220:RC$=""
540 FORK=1TOL(A):GET#1,A$:IFA$=CHR$(160)
    THENK=L(A)
550 RC$=RC$+A$:NEXTK:PRINT
560 PRINT#PR,"{SH R}EC.{SH N}."I:PRINT#P
    R,RC$:PRINT#PR:NEXTI
570 CLOSE1:RETURN
580 REM **RICERCA DI UN RECORD**
590 INPUT"{CLR}{SH C}AMPO CHIAVE";A$:CF=
    0:FORI=0TOTY:CT(I)=0:NEXT
600 PRINT"{CUR.GIU}{SH R}ICERCA IN CORSO
    ...{CUR.GIU}"
610 FORI=1TOTY:IFCC$(I)=""THENI=RN
620 IFA$=LEFT$(CC$(I),LEN(A$))THENCF=CF+
    1:CT(CF)=I
630 NEXT
640 IFCF=0THENPRINT"{CUR.GIU}{SH N}ESSUN
     RECORD CHE INIZI CON":PRINT"{RVS ON
    }"A$:GOSUB1420:RETURN
650 IFCF=1THEN710
660 PRINT"{CUR.GIU}{SH E}SISTONO"CF"RECO
    RD CHE INIZIANO CON":PRINT"{RVS ON}"
    A$
```

```
670 PRINT"{SH P}REMERE {SH C} PER VISUAL
    IZZARLI"
680 PRINT"{8 SPC}{SH E} PER USCIRE"
690 GOSUB1140:IFA<>69ANDA<>67THEN690
700 IFA=69THENRETURN
710 PRINT"{CLR}";:OPEN1,8,2,"0:"+F$:FORI
    =1TOCF:RC$="":PRINT#PR,"{SH R}ECORD
    {SH N}.";CT(I)
720 PRINT#PR
730 HB=INT(CT(I)/256):LB=CT(I)-HB*256
740 PO=1:GOSUB1220:PS=1:FORK=1TORL
750 GET#1,A$:RC$=RC$+A$:NEXT
760 FORK=1TOC:PRINT#PR,N$(K)": "MID$(RC$
    ,PS,L(K)):PRINT#PR:PS=PS+L(K):NEXT
770 PRINT"{CUR.GIU}{SH P}REMERE UN TASTO
     PER PROSEGUIRE":GOSUB1140:NEXTI:CLO
    SE1:RETURN
780 REM **SUBROUTINE INPUT CONTROLLATO *
    *
790 REM **PARAMETRI IN USCITA DI QUESTA
    ROUTINE**
800 REM **                     B$:STRINGA
    **
810 REM **           LS:LUNGHEZZA STRINGA
    **
820 REM **PARAMETRI IN ENTRATA DI QUESTA
     ROUTINE**
830 REM **           L(W):LUNGHEZZA CAMPO
    **
840 PRINT"{SH I}NSERISCI (TERMINA CON '{
    SH R}{SH E}{SH T}{SH U}{SH R}{SH N}'
    ):":PRINT:F=1:B$="":LS=0:A$="":A=0
850 GETA$:IFA$=""THENGOSUB940:GOTO850
860 A=ASC(A$+CHR$(0))
870 IFA$=CHR$(20)THENLS=LS-1:F=0:B$=LEFT
    $(B$,LEN(B$)-1):GOTO910
880 IFA$=CHR$(13)THENF=0:GOTO970
890 IF(A>31ANDA<95)OR(A>159ANDA<233)THEN
910
```

```
900 PRINT:PRINT"{SH C}{SH A}{SH R}{SH A}
    {2 SH T}{SH E}{SH R}{SH E} {SH N}{SH
     O}{SH N} {SH C}{SH O}{SH N}{SH S}{S
    H E}{SH N}{SH T}{SH I}{SH T}{SH O}!"
    :PRINT:GOTO840
910 PRINTA$;
920 IFF=1THENB$=B$+A$:LS=LS+1:IFLEN(B$)=
    L(W)THENRETURN
930 F=1:GOTO850
940 C1=PEEK(205)+1:C2=PEEK(202):L=3072+4
    0*(C1-1)+C2:IFL<3072ORL>4071THENRETU
    RN
950 POKEL-1024,127:POKEL,160:FORI=1TO80:
    NEXT:POKEL,32:FORI=1TO80:NEXT:RETURN
960 REM **AGGIUNGE SPAZI SHIFTATI FINO A
     LUNGHEZZA CAMPO**
970 A=LEN(B$):D=L(W)-A:IFA=0THENRETURN
980 IFD>0THEN1000
990 IFD<0THENB$=LEFT$(B$,L(W)):RETURN
1000 A$=CHR$(160):C$="":FORI=1TOD:C$=C$+
     A$:NEXT:B$=B$+C$:RETURN
1010 OPEN3,8,3,"0:I"+F$+",S,R":GOSUB1170
1020 IFE1$<>"00"THENGOSUB1420:RUN
1030 PRINT"{CUR.SU}{SH L}ETTURA FILE IND
     ICE...{CUR.SU}"
1040 INPUT#3,RN,RL,C,TY:TRAP2650:DIML(C)
     ,N$(C),CC$(RN),CT(RN)
1050 FORI=1TOC:INPUT#3,N$(I),L(I):NEXT
1060 FORI=1TOTY:INPUT#3,CC$(I):A=ASC(LEF
     T$(CC$(I),1)+CHR$(0))
1070 NEXT:CLOSE3
1080 PRINT#PR,"{CLR}{SH F}{SH I}{SH L}{S
     H E}:"F$TAB(20)"{RVS OFF}{SH N}.REC
     ORD:"RN
1090 PRINT#PR,"{CUR.GIU}{SH L}UNGH.RECOR
     D:"RLTAB(20)"{SH C}AMPI:"C"{CUR.GIU
     }"
1100 FORI=1TOC:PRINT#PR,"{SH C}AMPO";I;"
     :";N$(I)TAB(20)"{SH L}UNGH.:"L(I):N
     EXT
```

```
1110 PRINT#PR:PRINT#PR,"{SH R}ECORD ALLO
     CATI:"TY
1120 PRINT"{2 CUR.GIU}{SH P}{SH R}{SH E}
     {SH M}{SH E}{SH R}{SH E} {SH U}{SH
     N} {SH T}{SH A}{SH S}{SH T}{SH O} {
     SH P}{SH E}{SH R} {SH P}{SH R}{SH O
     }{SH S}{SH E}{SH G}{SH U}{SH I}{SH
     R}{SH E}":GOSUB1140:GOTO1240
1130 REM **SUBROUTINE ATTESA PRESSIONE D
     I UN TASTO**
1140 GETA$:IFA$=""THEN1140
1150 A=ASC(A$+CHR$(0)):RETURN
1160 REM **SUBROUTINE LETTURA CANALE ERR
     ORI DEL DRIVE**
1170 INPUT#15,E1$,E2$,E3$,E4$:PRINT"{CUR
     .GIU}ERRORI:";
1180 PRINTE1$;" {RVS ON}";:E3$=""
1190 FORI=1TOLEN(E2$):E3$=E3$+CHR$(ASC(M
     ID$(E2$,I,1))+128):NEXT:PRINTE3$
1200 RETURN
1210 REM **SUBROUTINE POSIZIONAMENTO PUN
     TATORE AL RECORD**
1220 PRINT#15,"P"+CHR$(2+96)+CHR$(LB)+CH
     R$(HB)+CHR$(PO):RETURN
1230 REM **MENU PRINCIPALE**
1240 PRINT"{CLR}{RVS ON}{SH R}{SH E}{SH
     L}{SH F}{SH I}{SH L}{SH I}{SH N}{SH
      G} {SH S}{SH Y}{SH S}{SH T}{SH E}{
     SH M} 1.0{14 SPC}{SH M}{SH C}{SH G}
      85"
1250 A$="{5 CUR.DES}":B$="{5 CUR.GIU}":P
     R=2
1260 PRINTB$;A$;"1. {SH C}REAZIONE NUOVO
      FILE"
1270 PRINTA$;"{CUR.GIU}2. {SH D}EFINIZIO
     NE FILE CORRENTE"
1280 PRINTA$;"{CUR.GIU}3. {SH L}ETTURA F
     ILE CORRENTE"
```

```
1290 PRINTA$;"{CUR.GIU}4. {SH S}CRITTURA
      FILE CORRENTE"
1300 PRINTA$;"{CUR.GIU}5. {SH R}ICERCA D
     I UN RECORD"
1310 PRINTA$;"{CUR.GIU}6. {SH O}PZIONI D
     I STAMPA"
1320 PRINTA$;"{CUR.GIU}7. {SH M}EMORIA L
     IBERA"
1330 PRINTA$;"{CUR.GIU}8. {SH F}INE OPER
     AZIONI"
1340 GOSUB1140:IFA<49ORA>56THEN1340
1350 ONA¬48GOTO1430,1380,2330,2080,1360,
     200,120,2540
1360 GOSUB590
1370 GOTO1240
1380 CLR:PRINT"{CLR}":GOSUB2730:OPEN2,3:
     PR=PEEK(819):IFPR=4THENOPEN4,4,7
1390 INPUT"{CUR.GIU}{SH N}OME DEL FILE C
     ORRENTE";F$:F$=CHR$(18)+F$:OPEN15,8
     ,15
1400 PRINT"{CUR.GIU}{SH O}{SH K}.":GOTO1
     010
1410 REM **SUBROUTINE CICLO DI ATTESA**
1420 FORI=1TO2000:NEXT:RETURN
1430 REM **INIZIALIZZAZIONE DEL PROGRAMM
     A**
1440 CLR:OPEN2,3:PR=PEEK(819):IFPR=4THEN
     OPEN4,4,7
1450 OPEN15,8,15:PRINT"{CLR}{SH I}NSERIS
     CI IL DISCO DATI E PREMI RETURN"
1460 PRINT"OPPURE PREMI SPAZIO PER GENER
     ARLO":PRINT
1470 GETA$:IFA$=""THEN1470
1480 IFA$=CHR$(13)THEN1640
1490 IFA$=CHR$(32)THEN1510
1500 GOTO1470
1510 PRINT"{CUR.GIU}{SH L}A CREAZIONE DE
     L DISCO DATI COMPORTA LA"
1520 PRINT"PERDITA DI TUTTI I DATI EVENT
     UALMENTE"
```

```
1530 PRINT"PRESENTI SUL DISCO STESSO."
1540 INPUT"{2 CUR.GIU}{SH S}{SH E}{SH I}
      {SH S}{SH I}{SH C}{SH U}{SH R}{SH
     O} ({SH S}/{SH N})";A$
1550 IFA$<>"S"THENRUN
1560 INPUT"{CUR.GIU}{SH N}OME DEL DATABA
     SE{4 SPC}";A$
1570 IFLEN(A$)>16ORLEN(A$)<1THENPRINT"{S
     H D}{SH A} 1 {SH A} 16 {SH C}{SH A}
     {SH R}{SH A}{2 SH T}{SH E}{SH R}{SH
      I}!":GOTO1560
1580 INPUT"{SH C}ODICE IDENTIFICATORE";B
     $
1590 IFLEN(B$)<>2THENPRINT"{SH C}{SH O}{
     SH D}{SH I}{SH C}{SH E} {SH D}{SH I
     } 2 {SH C}{SH A}{SH R}{SH A}{2 SH T
     }{SH E}{SH R}{SH I}!":GOTO1580
1600 PRINT#15,"NO:"+A$+","+B$
1610 PRINT"{CUR.GIU}{SH F}ORMATTAZIONE I
     N CORSO. {SH A}TTENDERE."
1620 GOSUB1170
1630 REM **LETTURA NUMERO DI BLOCCHI LIB
     ERI SUL DISCO**
1640 GOSUB2730:OPEN3,8,0,"$0:%&%":I=0
1650 FORI=1TO34:GET#3,A$:NEXT
1660 GET#3,A$:A=ASC(A$+CHR$(0))
1670 GET#3,A$:B=ASC(A$+CHR$(0))
1680 SC=(A+B*256)-6:PRINT"{SH B}LOCCHI L
     IBERI PER I DATI:"SC
1690 CLOSE3
1700 GOSUB1170
1710 REM **DEFINIZIONE DEL FILE**
1720 INPUT"{SH Q}UANTI CAMPI PER RECORD"
     ;C:PRINT:TRAP2650:DIMN$(C),L(C)
1730 RL=0:FORI=1TOC
1740 PRINT"{RVS ON}{SH C}AMPO N.{RVS OFF
     }"I:PRINT
1750 INPUT"{SH N}OME";N$(I)
```

```
1760 INPUT"{SH L}UNGHEZZA";L(I):IFI=1AND
     L(I)>79THENPRINT"{SH M}{SH A}{SH X}
     .79 {SH S}{SH U}{SH L} {SH P}{SH R}
     {SH I}{SH M}{SH O} {SH C}{SH A}{SH
     M}{SH P}{SH O}!":GOT
1770 RL=RL+L(I):PRINT:NEXT
1780 RL=RL+1:PRINT"{SH L}UNGHEZZA TOTALE
      DI UN RECORD:":PRINTRL"CARATTERI."
     :PRINT
1790 IFRL>254THENPRINT"{SH R}{SH E}{SH C
     }{SH O}{SH R}{SH D} {SH T}{SH R}{SH
      O}{2 SH P}{SH O} {SH L}{SH U}{SH N
     }{SH G}{SH H}{SH I}!":FORI=1TO5000:
     NEXT:RUN
1800 REM **CREA IL FILE RELATIVO**
1810 RN=0:PRINT"{SH D}IMENSIONI DEL FILE
      ({SH N}.RECORD)";:INPUTRN:RN=RN+1:
     PRINT
1820 IFRN*RL/254<=SCTHEN1850
1830 PRINT"{CLR}{SH F}{SH I}{SH L}{SH E}
     {SH SPC}{SH T}{SH R}{SH O}{2 SH P}{
     SH O}{SH SPC}{SH G}{SH R}{SH A}{SH
     N}{SH D}{SH E}{SH SPC}{SH P}{SH E}{
     SH R}{SH SPC}{SH E}{2 SH S}{SH E}{S
     H R}{SH E} {SH C}{SH O}{SH N}{SH T}
     {SH E}{SH N}{SH U}{SH T}{SH O}"
1840 PRINT"{SH S}{SH U} {SH Q}{SH U}{SH
     E}{SH S}{SH T}{SH O} {SH D}{SH I}{S
     H S}{SH C}{SH O}!":GOTO1810
1850 TRAP2650:DIMCC$(RN),CT(RN)
1860 INPUT"{SH N}OME DEL FILE";F$:IFLEN(
     F$)>0ANDLEN(F$)<14THENF$=CHR$(18)+F
     $:GOTO1890
1870 PRINT"{SH D}{SH A} 1 {SH A} 13 {SH
     C}{SH A}{SH R}{SH A}{2 SH T}{SH E}{
     SH R}{SH I}!":GOTO1850
1880 REM **CREAZIONE DEL FILE RELATIVO S
     U DISCO**
1890 PRINT#15,"I0"
```

```
1900 PRINT"{CLR}{SH C}REAZIONE FILE INDI
     CE...":TY=0
1910 OPEN3,8,3,"@0:I"+F$+",S,W":GOSUB117
     0
1920 R$=CHR$(13):PRINT#3,RN;R$;RL;R$;C;R
     $;TY
1930 FORI=1TOC:PRINT#3,N$(I);R$;L(I):NEX
     T
1940 FORI=1TOTY:IFCC$(I)=""THENCC$(I)=CH
     R$(255)
1950 PRINT#3,CC$(I):NEXT
1960 CLOSE3
1970 HB=INT(RN/256)
1980 LB=RN-HB*256
1990 PRINT"{2 CUR.GIU}{SH C}REAZIONE FIL
     E DATI..."
2000 OPEN1,8,2,"0:"+F$+",L,"+CHR$(RL)
2010 PO=1:GOSUB1220
2020 PRINT#1,CHR$(255)
2030 PO=1:GOSUB1220
2040 CLOSE1
2050 GOSUB1170:IFE1$<>"50"ANDE1$<>"00"TH
     ENEND
2060 GOSUB1420:GOTO1240
2070 REM **PREPARA PER LA SCRITTURA**
2080 R=TY+1
2090 PRINT"{CLR}{RVS ON}+ {SH S}{SH U}{2
      SH C}. - {SH P}{SH R}{SH E}{SH C}.
      \ {SH S}{SH I}{SH N}{SH G}. {SH E}
      {SH E}{SH S}{SH C}{SH E}{2 SPC}{SH
      I} {SH I}{SH N}{SH S}{SH E}{SH R}.
     "
2100 PRINT#PR,"{SH R}{SH E}{SH C}{SH O}{
     SH R}{SH D} #"R
2110 PRINT"{CUR.GIU}{SH S}CELTA?"
2120 GOSUB1140:IFA<>43ANDA<>45ANDA<>92AN
     DA<>69ANDA<>73THEN2120
2130 IFA=43THENR=R+1:GOTO2090
2140 IFA=45THENR=R-1:IFR<1THENR=1
```

```
2150 IFA=45THEN2090
2160 IFA=92THENINPUT"{SH N}. RECORD";R:G
     OTO2090
2170 IFA=69THENCLOSE1:GOTO1240
2190 RC$="":FORW=1TOC:PRINT#PR:B$=""
2200 PRINTN$(W)"("L(W)"CARATT.):":GOSUB8
     40:IFW=1THENCC$(R)=B$
2210 RC$=RC$+B$
2220 NEXT
2230 REM **SCRIVE SUL FILE**
2250 OPEN1,8,2,"0:"+F$
2260 HB=INT(R/256)
2270 LB=R¬HB*256
2280 PO=1:GOSUB1220
2290 PRINT#1,RC$:PO=1:GOSUB1220
2300 CLOSE1:GOSUB1170:R=R+1:IFCC$(R)<>""
     THEN2090
2310 TY=TY+1:GOTO2090
2320 REM **LEGGE SEQUENZIALMENTE IL FILE
      RELATIVO**
2330 R=1
2340 PRINT"{CLR}{RVS ON}+ {SH S}{SH U}{2
      SH C}.{3 SPC}¬ {SH P}{SH R}{SH E}{
     SH C}.{4 SPC}\ {SH S}{SH I}{SH N}{S
     H G}.{3 SPC}{SH E} {SH E}{SH S}{SH
     C}{SH E}{3 SPC}"
2350 RC$="":PRINT#PR,"{SH R}{SH E}{SH C}
     {SH O}{SH R}{SH D} #";R:PRINT#PR
2360 IFR>TYTHENPRINT"{SH R}{SH E}{SH C}{
     SH O}{SH R}{SH D} {SH V}{SH U}{SH O
     }{SH T}{SH O}!":GOSUB1420:GOTO2460
2370 HB=INT(R/256)
2380 LB=R-HB*256
2390 OPEN1,8,2,"0:"+F$
2400 PO=1:GOSUB1220:PS=1:FORI=1TORL
2410 GET#1,A$:RC$=RC$+A$:NEXTI:CLOSE1
2420 FORK=1TOC:PRINT#PR,N$(K)": "MID$(RC
     $,PS,L(K)):PRINT#PR:PS=PS+L(K)
2430 IFK<>1THEN2450
```

```
2440 CC$(R)=LEFT$(RC$,L(1))
2450 NEXT:PRINT#PR:GOSUB1170
2460 PRINT"{CUR.GIU}{SH S}CELTA?":GOSUB1
     140
2470 IFA<>43ANDA<>45ANDA<>92ANDA<>69THEN
     GOSUB1140:GOTO2470
2480 IFA=43THENR=R+1:GOTO2340
2490 IFA=45THENR=R-1:IFR<1THENR=1
2500 IFA=45THEN2340
2510 IFA=92THENINPUT"{SH N}. RECORD";R:G
     OTO2340
2520 IFA=69THENCLOSE1:GOTO1240
2530 REM **RISCRITTURA FILE INDICE**
2540 PRINT"{CLR}{SH S}CRITTURA FILE INDI
     CE..."
2550 IFTY>RNTHENRN=TY
2560 OPEN3,8,3,"@0:I"+F$+",S,W":GOSUB117
     0
2570 R$=CHR$(13):PRINT#3,RN;R$;RL;R$;C;R
     $;TY
2580 FORI=1TOC:PRINT#3,N$(I);R$;L(I):NEX
     T
2590 FORI=1TOTY:IFCC$(I)=""THENCC$(I)=CH
     R$(255)
2600 PRINT#3,CC$(I):NEXT:CLOSE3
2610 INPUT"{SH F}INE DEFINITIVA DEI LAVO
     RI (S/N){2 SPC}N{3 CUR.SIN}";A$
2620 IFA$<>"S"THENRUN
2630 SYS32768
2640 REM **INTERCETTAZIONE ERRORI**
2650 PRINT"{RVS ON}{SH A}{2 SH T}{SH E}{
     SH N}{SH Z}{SH I}{SH O}{SH N}{SH E}
     ! {SH S}I E' VERIFICATO UN ERRORE!"
2660 PRINT"{CUR.GIU}{SH E}RRORE:"ER:PRIN
     T"{CUR.GIU}{SH T}IPO: "ERR$(ER)
2670 IFER<>16THEN2710
2680 PRINT"{CUR.GIU}{SH O}CCORRE DIMINUI
     RE IL NUMERO DEI RECORD"
```

```
2690 PRINT"OPPURE IL FILE E' TROPPO GRAN
     DE PER"
2700 PRINT"ESSERE GESTITO DA QUESTO COMP
     UTER."
2710 GOSUB 1420:RESUME110
2720 REM **VISUALIZZAZIONE FILE DATI PRE
     SENTI SUL DISCO**
2730 P$=CHR$(34):OPEN8,8,0,"$0":PRINT"{C
     UR.GIU}{SH A}{SH N}{SH A}{SH L}{SH
     I}{SH S}{SH I} {SH D}{SH I}{SH S}{S
     H C}{SH O} {SH D}{SH A}{SH T}{SH I}
     :{CUR.GIU}":GOSUB2790
2740 J=1:GOSUB2790:F$="":EF=0
2750 GOSUB2790
2760 IFEF=0THENJ=J+1:F$="":GOTO2750
2770 CLOSE8:IFW=0THENPRINT"{SH N}ESSUN F
     I&E DATI SU QUESTO DISCO.":RETURN
2780 PRINT"{SH F}ILE DATI PRESENTI:";W:R
     ETURN
2790 GET#8,A$:IFA$<>N$THEN2790
2800 GET#8,A$:IFA$<>N$THEN2830
2810 GET#8,A$:IFA$<>N$THEN2830
2820 GET#8,A$:IFA$=N$THENEF=1:RETURN
2830 IFA$<>P$THEN2800
2840 GET#8,A$:IFA$=P$THEN2860
2850 F$=F$+A$:GOTO2840
2860 IFJ=0THENPRINT"{SH N}OME DEL {SH D}
     ATABASE: ";F$
2870 IFLEFT$(F$,1)=CHR$(18)ANDLEFT$(F$,2
     )<>"I"+CHR$(18)THENPRINT"{SH F}ILE
     DATI:";F$:W=W+1
2880 RETURN
```

2.5 I database sequenziali

La limitazione posta dall'utilizzo forzato del registratore a cassette, impone la programmazione del software per archivio nella forma sequenziale.
Il termine 'sequenziale', già di per sè esemplicativo, indica che i dati vengono

letti o scritti in sequenza: per vedere che cosa contiene il centesimo record, siamo obbligati a leggere anche tutti i precedenti novantanove, malgrado il loro contenuto non ci interessi.
Inoltre, se disponiamo di un file con cento record e desideriamo aggiungervi il centounesimo, non sarà possibile — come avviene per i file relativi su disco — 'incollare' questo record alla fine del file, ma dovremo riscrivere per intero tutto il file.
Risulta evidente che i file sequenziali sono molto più lenti dei relativi e, se a questa lentezza sommiamo quella insita nelle operazioni con il registratore a cassette, il tempo necessario al trattamento di file mediamente lunghi sale notevolmente.
Fortunatamente esistono particolari tecniche che permettono di ridurre notevolmente il tempo di accesso al singolo record — fino a renderlo inferiore a quello dei file relativi — malgrado il tempo *globale* di accesso al file rimanga inevitabilmente superiore.
Queste tecniche consistono nel caricare in RAM l'*intero* file all'inizio delle operazioni, quindi manipolarlo con gran velocità (tutte le operazioni che avvengono in RAM hanno una velocità enormemente superiore a qualsiasi operazione anche su disco), infine risalvare su nastro il file manipolato.
I lunghi tempi necessari all'iniziale caricamento del file ed al suo finale salvataggio, sono compensati dalla velocità delle operazioni sul file, siano esse l'aggiunta di un record, la sua ricerca, od un ordinamento alfabetico.
Il file esistente su nastro viene trasferito nella RAM del computer, all'interno di opportuni vettori dimensionati su un valore massimo, che rappresenta la capacità del file (massimo numero di record). Il contenuto di questi vettori può essere quindi alterato e manipolato in qualsiasi modo attraverso le opzioni del database, aggiungendo, eliminando o modificando qualsiasi record. Alla fine delle operazioni si potrà quindi risalvare il tutto su nastro.
Poiché, come abbiamo detto, le operazioni più costose in termini di tempo sono quelle con l'unità a cassette, risulta assai conveniente caricare il file all'inizio dell'attività — generalmente al mattino — e procedere al suo salvataggio solo alla sera al termine dell'utilizzo del database. Per tutta la giornata lavorativa disporremo così di un archivio estremamente veloce e funzionale.

2.6 La programmazione

I file sequenziali sono molto più facili da gestire rispetto ai file relativi: scrivere su nastro dei file dati è molto semplice: basta aprire un canale di scrittura tra il computer ed il registratore (OPEN), scrivere nel file le variabili (PRINT#), e chiudere il canale (CLOSE). Si usa l'istruzione PRINT#, in quanto essa restituisce il valore di una variabile indipendentemente dal nome che le abbiamo assegnato: se A= 5 ed X$= "CIAO", PRINTA visualizzarà 5 e PRINTX$ visualizzerà CIAO; idem per PRINT#, con la sola differenza che invece di visualizzare il valore delle variabili, le invia sul canale di scrittura. Sono tutte qui le istruzioni occorrenti per la scrittura di un file di dati.

Al contrario, per leggere un file dati dall'unità a cassette, si dovrà aprire un canale di lettura (OPEN), assegnare i valori contenuti nel file ad alcune variabili (INPUT# o GET#), e chiudere il canale (CLOSE). Con un'analogia, i file dati sostiuiscono il registratore allo schermo nell'istruzione PRINT, ed ancora il registratore alla tastiera (cioè voi stessi) nell'istruzione INPUT o GET. Se terrete sempre presente questo paragone, vi sarà molto più facile ed intuitivo capire i criteri di manipolazione dei file dati.
Poniamo il caso si voglia avere un file dati che contenga esclusivamente stringhe. Il programma che scrive il file sarà ad esempio il seguente:

```
10 OPEN1,1,1, "FILEDATI"
20 A$= "CIAO":B$= "HALLO":C$ = "ALOHA"
30 PRINT#1,A$,B$,C$
40 CLOSE1
50 END
```

Il maggior problema che, in generale, è sentito dai programmatori alle prime armi non è tanto scrivere i file, quanto leggerli correttamente. Per il momento non preoccupatevi dei dettagli, e guardate la linea 30: immaginate che la PRINT#1 agisca sullo schermo. Il risultato sarebbe:

```
CIAO          HALLO          ALOHA
```

Ci saranno cioè degli spazi tra le stringhe, dovuti alla virgola che le separa nell'istruzione PRINT. Supponiamo ora che vogliate "rileggere" queste stringhe; usereste un'istruzione del tipo INPUTA$,B$,C$. Che cosa accadrebbe? Se, alla comparsa del cursore della INPUT, digitaste le tre parole con tutti gli spazi tra loro (così come appaiono sullo schermo), esse verrebbero tutte e tre, insieme agli spazi, immagazzinate nella variabile A$. Questo perché, come ben sapete, in una INPUT multipla ogni singola parte deve terminare con un RETURN, oppure bisogna battere delle virgole per separarne le varie parti. Se cambiassimo quindi la linea 30 in questo modo:

```
30 PRINT#1, A$+","+B$+","+C$
```

Il file verrà scritto come (ricordate l'analogia con la INPUT sullo schermo):

```
CIAO, HALLO, ALOHA
```

Proprio come le avreste digitate in un istruzione INPUT.
Potremmo anche separare le varie stringhe con un RETURN, usando varie istruzione PRINT:

```
30 PRINT#1,A$:PRINT#1,B$:PRINT#1,C$
```

Se stampate sullo schermo, il risultato sarebbe:

```
CIAO
HALLO
ALOHA
```

Tenendo a mente che il CHR$(13) equivale ad un RETURN, potremmo raggruppare le variabili in questo modo:

```
30 PRINT#1,A$+CHR$(13)+B$+CHR$(13)+C$
```

Non mettiamo un CHR$(13) dopo l'ultima variabile in quanto esso viene posto automaticamente dal computer alla fine di ogni linea.
Ancora, abbiate sempre in mente come apparirebbero le variabili se stampate sullo schermo: occorrerà sempre scriverle in modo tale che una INPUT possa leggerle in maniera appropriata.
Passiamo ai dettagli. Per i file dati si apre un canale con il registratore con la sintassi richiesta dall'istruzione OPEN:

```
10 OPEN1,1,1,"FILEDATI"
```

In questa istruzione, 1 è il numero di canale che abbiamo deciso di usare, il secondo 1 è il numero di periferica del registratore, ed il terzo 1 rappresenta l'indirizzo secondario e stabilisce che desideriamo *scrivere* dei dati e non leggerli.
Quando si scrivono o leggono dati da un file, questi non vengono scritti sul nastro (durante un output) o letti nel computer (durante un input) fino a che il buffer del registratore (locazioni 819-1010) non si sia riempito con 191 caratteri. Solo in questo momento il contenuto del buffer viene inviato al computer o scritto sul nastro magnetico. Naturalmente è possibile scrivere un programma con la seguente forma:

```
10 INPUT"NOME DEL FILE";F$
15 OPEN 1,1,1,F$
```

Dopo aver aperto il file per la scrittura, dovremo inviare qualcosa all'unità a cassette. Ecco qualche esempio:
Per salvare un vettore numerico a 10 elementi:

```
29 FORI=1T010
30 PRINT#1,A(I)
40 NEXT
```

Per salvare una matrice bidimensionale di stringa tre per sei:

```
20 FORI=1TO3
30 FORJ=1TO6
40 PRINT#1,A$(I,J)
50 NEXTJ,I
```

Dopo aver inviato al registratore tutti i dati desiderati, occorre chiudere il file, e per due ragioni. In primo luogo, può darsi che il programma riutilizzi in seguito lo stesso numero di canale, e la CLOSE lo rende nuovamente disponibile; in secondo luogo, se il canale non viene chiuso, il contenuto dell'ultimo buffer potrebbe non venire scritto sul nastro — ricordate che il contenuto del buffer non viene passato al registratore fino a che non ammonti a 191 byte — e quando finite di inviare dati può darsi (è anzi molto probabile) che il contenuto del buffer non sia esattemente di 191 byte. Una CLOSE provoca il passaggio dei dati al registratore anche se il buffer non è pieno.

La CLOSE va seguita dal numero il canale (il primo numero usato nella OPEN).

Come accennato prima, sembra che le difficoltà maggiori si abbiano nella lettura dei file piuttosto che nella scrittura. Un errore comune consiste nel leggere i valori fuori ordine, oppure cercare di leggere un valore di stringa in una variabile numerica. Se tenete presente la filosofia di funzionamento dei comandi PRINT ed INPUT non avrete problemi di questo genere. Questa linea aprirà un file chiamato "FILEDATI" che sia stato precedentemente scritto sul nastro:

```
100 OPEN 1,1,0,"FILEDATI"
```

Potete usare l'istruzione INPUT# per leggere il file nello stesso modo con il quale era stato scritto:

```
110 INPUT#1,A$
```

Dal momento che si leggono valori e non variabili, si può usare qualsiasi nome valido di variabile.
Ecco alcuni esempi di INPUT# paralleli a quelli presentati per PRINT#. Se seguite specularmente la forma usata nella PRINT#,non potete sbagliare:
Legge un vettore numerico di 10 elementi:

```
120 FORI=1TO10
130 INPUT#1,A(I)
140 NEXT
```

Legge una matrice di stringa bidimensionale tre per sei:

```
120 FORI=1T03
130 FORJ=1T06
140 INPUT#1, B$(I,J)
150 NEXTJ,I
```

Come avete visto, non è necessario usare gli stessi nomi di variabile, ma solo rispettare l'ordine di sequenza dei valori.
Quando avete finito di leggere il file, ricordatevi di chiuderlo.
Passiamo ora ad esaminare qualche tecnica di gestione dei file dati. I nostri esempi fino ad ora sono stati estremamente semplici, assumendo come dato di fatto un numero fisso di record nel file.
Ma, se vi capita di dover leggere un file senza sapere quanto sia lungo, come fare a capire quando è ora di fermare il ciclo di lettura?

```
10 INPUT"NUMERO DEI NOMI";N
20 DIMA$(N)
30 FORI=1TON
40 PRINT"NOME N.";I;":";
50 INPUTA$(I)
60 NEXTI
70 OPEN1,1,1,"FILEDATI"
80 FORI=1TON
90 PRINT#1,A$(I)
100 NEXTI
110 CLOSE1
```

Questo programma chiede all'utente una lista di nomi ed il numero di questi ultimi, quindi li registra su nastro in un file chiamato FILEDATI.
Ora, volete leggere questo file con un altro programma, e non vi ricordate più quanti nomi sono stati scritti nel file. La soluzione: scrivete N, il numero di nomi, nel file! Aggiungiamo quindi questa linea:

```
75 PRINT#1,N
```

Ora siamo in grado di scrivere il semplicissimo programma di lettura:

```
10 OPEN1,1,0,"FILEDATI"
20 INPUT#1,N
30 DIMA$(N)
40 FORI=1TON
50 INPUT#1,A$(I)
60 NEXTI
70 CLOSE1
```

Un altro sistema consiste nello scrivere uno speciale separatore di fine file, poniamo un asterisco, in fondo al file. E c'è ancora un altro metodo: il computer cambia il valore della variabile riservata STATUS (ST) — che normalmente vale 0 — quando trova un errore in una operazione di I/O. Uno di questi errori è il voler continuare a leggere oltre la fine del file. Se non avessimo aggiunto al programma la linea 75, potremmo comunque leggere il file con un programma come questo:

```
10 OPEN1,1,0,"FILEDATI"
20 DIMA$(50):N=1
30 INPUT#1,A$(N)
40 IF ST =0 THENN=N+1:GOTO30
50 CLOSE 1
```

Se inoltre volete vedere visualizzati i dati mano a mano che vengono letti, basta aggiungere una banale linea come questa:

```
35 PRINTA$(N)
```

Si può anche usare l'istruzione GET# per leggere un file. Il programma che segue visualizza qualsiasi possibile file sequenziale sullo schermo:

```
10 OPEN1,1,0,"NOME DEL FILE"
20 GET#1,A$:S=ST:PRINTA$;
30 IFS=0THEN20
40 CLOSE1
```

Ora può essere che si desideri leggere un file come sequenza di numeri ASCII. Sostituite la linea 20:

```
20 GET#1,A$:S=ST:A=ASC(A$)
```

Prima di procedere, una precisazione: se il file contiene qualche stringa nulla, otterrete un "ILLEGAL QUANTITY ERROR" in quanto la funzione ASC non accetta come argomento stringhe nulle. Dovrete quindi scrivere la linea 20 in questo modo:

```
20 GET#1,A$:S=ST:A=ASC(A$+CHR$(0)):PRINTA
```

La funzione ASCII restituirà ora il valore 0 in caso di stringhe nulle, senza alcun messaggio d'errore.
I comandi per la gestione dei file sequenziali sono quindi molto semplici e di uso intuitivo, essendo del tutto simili alle familiari istruzioni PRINT, INPUT e GET che utilizziamo normalmente nei nostri programmi.

2.7 Seqfiling system 1.0: un database sequenziale per sistemi a cassette

La struttura programma/utente è stata mantenuta simile a quella del database relativo presentato nei paragrafi precedenti di questo capitolo, per avere una certa uniformità nei comandi del programma.
All'attivazione del programma, viene richiesto di inserire la data corrente nella forma giorno/mese/anno (esempio 011186), e l'ora corrente nella forma ore/minuti/secondi (esempio 110800). Se non si desidera — per brevità — inserire alcun valore, si prema semplicemente RETURN ai prompt del programma; in questo caso tuttavia non verranno registrate data ed ora dell'ultimo aggiornamento effettuato sul file dati.
Comparirà quindi un menu di selezione contenente le seguenti opzioni:

0. Crea un nuovo file
1. Carica file esistente
2. Salva file in memoria
3. Scrive un record
4. Legge un record
5. Ricerca di un record
6. Ordina alfabeticamente
7. Memoria libera
8. Stampa il file
9. Fine operazioni

Dal momento che alla prima utilizzazione non disporremo di un file dati su nastro, dovremo generarlo. Selezioniamo quindi l'opzione 0.
Il programma chiederà il numero di campi per record, e quindi il loro nome e rispettiva lunghezza (Per la trattazione su campi, record e file si vedano i paragrafi 2.1 e 2.2 di questo stesso capitolo). Viene quindi richiesta la capacità del file (massimo numero di record): qui occorre prestare molta attenzione. Non esiste un tetto massimo a questo valore, in quanto esso dipende dal numero di campi che abbiamo stabilito precedentemente: 100 record con 5 campi ciascuno occupano più memoria di 80 record con 15 campi ciascuno... Dando un valore di 1000 alle dimensioni del file si dovrebbe rientrare nella disponibilità di memoria del C16 inespanso, se ogni record è dotato di cinque o sei campi. Ad ogni buon conto, dopo aver assegnato le dimensioni al file e quindi il suo nome, è conveniente controllare tramite l'opzione 7 del menu l'ammontare di memoria libera: quest'ultima non dovrebbe mai essere inferiore a circa 1000 byte, al fine di lasciare il sufficiente spazio per le variabili utilizzate dal programma. Dopo un minimo di esperienza si sarà in grado di ottimizzare la capacità dei file sul massimo valore possibile.
Stabilite le massime dimensioni del file, dovremo scegliere un nome da assegnargli: chiamandolo "PROVA".
A questo punto potremo iniziare ad inserire informazioni attraverso l'opzione 3:

comparirà nella parte alta dello schermo una linea in campo inverso contenente le opzioni usabili in modo inserimento dati. Esse sono:

tasto "–" passa al record precedente
tasto "+" passa al record successivo
tasto "£" passa ad un record particolare
tasto "E" ritorna al menu principale
tasto "I" permette l'inserimento dati

A questo punto saremo posizionati sul record 1. Premiamo il tasto I per l'inserimento dati. Comparirà il nome del primo campo con tra parentesi la sua lunghezza, e saremo pronti per scrivere i nostri dati. Se, scrivendo, raggiungiamo la fine del campo, il computer passerà automaticamente al campo successivo; se invece i dati che desideriamo inserire hanno un lunghezza inferiore alla massima lunghezza del campo, dovremo premere RETURN per passare al campo successivo.
Se non si desidera scrivere su un campo, è sufficiente premere RETURN. Dopo aver inserito i dati in tutti i campi, il programma si posizionerà sul record successivo, in questo caso il numero 2.
Un record può essere riscritto in qualsiasi momento, posizionandosi su di esso con i tasti "+" e "—" oppure con il tasto '£' (lira) che permette di scrivere su un qualunque record comunicandone il numero al computer. Premendo quindi il tasto "I" si potrà riscrivere il record.
Scriviamo anche i dati del record 2 e ritorniamo al menu principale premendo il tasto "E".

Il nostro file conterrà ora due record, che potremo leggere con l'opzione 4. La schermata che compare è analoga a quella di scrittura (tranne per la parte alta allo schermo che contiene, in rosso, i dati generali del file), ed identiche sono le funzioni dei tasti tranne il tasto "I" che, ovviamente, non compare in questa opzione. Il programma si posizionerà automaticamente sul primo record e lo visualizzarà sullo schermo. Potremo leggere il secondo sia premendo "+" che premendo "£" e rispondendo con 2 alla domanda del computer. Dal momento che abbiamo inserito soltanto due record, cercando di leggere un record non scritto, ad esempio il numero 3, il computer ci avviserà che tale record è vuoto.
Premendo "E" ritorniamo al menu principale.
Una delle caratteristiche più utili dei programmi di archivio, è la possibilità di ricercare un certo record all'interno di un file che magari ne contiene più di mille.
L'opzione di ricerca esegue appunto questa funzione.
Il computer — selezionando tale opzione — visualizzerà l'elenco dei campi di ogni record e ci chiederà in base a quale campo desideriamo effettuare la ricerca.

Volendo, ad esempio, ricercare all'interno di un file di indirizzi composto da 5 campi (Cognome, Nome, Via, Città, Numero telefonico) il record contenente i dati del signor Rossi, sceglieremo di effettuare la ricerca sul primo campo, e digiteremo

"Rossi" alla richiesta circa il contenuto del campo. Il proogramma — dopo averci chiesto se desideriamo o meno una stampa dei dati — ricercherà quindi, all'interno del file, tutte le occorrenze del cognome "Rossi" nel primo campo. Se questo cognome compare nell'archivio, verrà visualizzato il record contenente i suoi dati.

Il programma chiederà quindi se si desidera proseguire nella ricerca (ci possono essere più persone il cui cognome è Rossi). In caso affermativo, la ricerca prosegue fino all'occorrenza successiva od alla fine del file. Se invece si desiderasse controllare tutte le persone che abitano a Milano, si sceglierà per la ricerca il campo numero 4 e gli si assegnerà come contenuto la stringa "Milano".

Una particolarità interessante dell'opzione di ricerca, è che essa permette di ricercare un particolare record semplicemente digitando solo una parte del suo campo chiave. Per spiegarci meglio, se il file contiene un record per il signor Rossi, uno per il signor Rossetti, ed uno per il signor Rosellini, ricercando "Rossi" verrà trovato solo il record del signor Rossi. Se, invece, alla domanda del computer sul contenuto del campo di ricerca degitiamo "Ross", il programma troverà sia "Rossi" che "Rossetti". Se digitiamo "R" verranno trovati tutti i record che iniziano per R.

Questa particolarità della funzione di ricerca risulta estremamente utile se non ricordiamo esattamente un cognome, o la forma del dato che ci serve. Ritorniamo al menu principale. L'opzione 7 ci dice l'ammontare della memora disponibile (dopo un tempo di attesa più o meno lungo necessario al computer per effettuare i suoi controlli).

Il programma è comunque dotato di un controllo interno, che segnala condizioni di errore generate dalla mancanza di memoria libera, nel caso il file si estenda in misura troppo elevata per essere gestita dalla memoria del C16.
L'opzione 6 ordina invece alfabeticamente tutto il file, a partire dal primo e poi via via per tutti i campi successivi, fino all'ultimo carattere di ogni record. Il tempo di attesa per l'operazione di ordinamento dipende dal numero esistente di record, dalla loro lunghezza, e dal loro stato di 'disordine' all'interno del file. Non è mai tuttavia troppo elevato, dal momento che tutta l'operazione avviene in RAM.
L'opzione 8 genera uno stampato di tutto il file, con l'ordine secondo il quale i record si trovano all'interno del file stesso. Ricordiamo che la stampa di record singoli è disponibile attraverso l'opzione di ricerca.

Infine, le opzioni 1 e 2 servono rispettivamente per il caricamento od il salvataggio su nastro di un file dati. Viene in entrambi i casi richiesto il nome del file e, mentre in caso di caricamento il bordo dello schermo diventa di colore porpora, in caso di salvataggio esso passa all'azzurro.

Si può uscire dal programma attraverso l'opzione 9 che, dopo aver richiesto conferma, provvede ad un parziale reset del computer.

Ecco ora un elenco delle principali variabili utilizzate dal programma:

RN	massimo numero di record
TY	numero di record allocati
F$	nome del file dati
RL	lunghezza dei record
C	numero dei campi
N$ ()	nome dei campi
L ()	lunghezza dei campi
RC$ ()	contenuto dei record
R$	ritorno carrello
BY	ammontare di byte liberi
F	variabile di flag
A	codice ASCII di un tasto
D$, U$	data aggiornamento
TI$, T$	ora aggiornamento
R	contatore dei record
PS	posizione di una sottostringa

2.8 Commento al listato

100-110 inizializzazione colori schermo, bordo e testo e passaggio al modo maiuscolo/minuscolo.

130-160 accettazione data ed ora corrente. A causa delle istruzioni CLR presenti nel programma, la data viene immagazzinata permanentemente nelle tre locazioni di memoria 1630, 1631 e 1632 nel suo formato rispettivamente di giorno, mese ed anno.

180-350 subroutine di input controllato per l'inserimento dei dati. Viene inoltre generato uno pseudo-cursore per facilitare l'inserimento stesso.

370-480 definizione del file dati. Si definiscono il numero di campi per record, il loro nome e lunghezza, e la capacità ed il nome del file. La massima lunghezza consentita di un record è di 255 caratteri (254 effettivi, in quanto il programma aggiunge automaticamente un CHR$(13) di fine record).

490-500 subroutine di attesa della pressione di un tasto. Viene restituito anche il codice ASCII del tasto premuto nella variabile A.

520-700 subroutine di inserimento dati. Questa subroutine permette sia l'inserimento sequenziale dei record, che il loro inserimento casuale all'interno del file, posto che non esistano altri record ancora vuoti dotati di numero progressivo inferiore.

720-840 subroutine di lettura dei record. Anch'essa permette sia la lettura sequenziale che quella di un record qualunque all'interno del file.

869-920 subroutine di intercettazione degli errori senza perdita dei dati.

940-970 subroutine di visualizzazione dei parametri principali del file durante la lettura dei record.

990-1070 subroutine di salvataggio su nastro dell'intero file.

1090-1170 subroutine di caricamento da nastro di un file, e di automatico dimensionamento dei vettori del programma.

1180-1320 menu di selezione delle opzioni offerte dal programma.

1340-1490 ricerca di un certo record all'interno del file, in base al contenuto di uno qualsiasi dei suoi campi. La ricerca avviene con il principio della 'wild card'.

1510-1580 subroutine di ordinamento sequenziale ed alfabetico del file. Vengono confrontati tra loro tutti i caratteri costituenti i vari record, e posti in ordine strettamente alfabetico.

1600-1630 subroutine di stampa di tutti i record contenuti nel file.

1640-1650 linea usata dalla routine di intercettazione degli errori, per effettuare un CLR e riprendere l'esecuzione del programma a partire dalla linea 1180.

1670-1680 visualizzazione dell'ammontare della memoria libera. È opportuno che questo valore non scenda mai al di sotto di 1000, a meno che non manchino ormai che pochi record al raggiungimento della massima capacità del file.

1700-1710 fine delle operazioni. Il programma chiede conferma e, in caso affermativo, provvede ad un parziale reset del computer.

SEQFILING SYSTEM 1.0

```
100 REM **INIZIALIZZAZIONE GENERALE**
110 TRAP850:COLOR0,1:COLOR4,1:COLOR1,2:P
    RINTCHR$(14)
120 REM **LA DATA VIENE IMMAGAZZINATA NE
    LLE LOCAZIONI 1630, 1631 E 1632**
130 INPUT"{CLR}{SH D}ATA CORRENTE ({2 SH
     G}{2 SH M}{2 SH A}){2 SPC}000000{8
    CUR.SIN}";D$
140 I=VAL(LEFT$(D$,2)):POKE1630,I:I=VAL(
    MID$(D$,3,2)):POKE1631,I
150 I=VAL(RIGHT$(D$,2)):POKE1632,I
160 INPUT"{CUR.GIU}{SH O}RA CORRENTE{2 S
    PC}({2 SH H}{2 SH M}{2 SH S}){2 SPC}
    000000{8 CUR.SIN}";TI$:GOTO1180
170 REM **SUBROUTINE INPUT CONTROLLATO *
    *
180 PRINT"{SH I}NSERISCI (TERMINA CON '{
    SH R}{SH E}{SH T}{SH U}{SH R}{SH N}'
    ):":PRINT:F=1:B$="":LS=0:A$="":A=0
190 GETA$:IFA$=""THENGOSUB280:GOTO190
200 A=ASC(A$+CHR$(0))
210 IFA$=CHR$(20)THENLS=LS-1:F=0:B$=LEFT
    $(B$,LEN(B$)-1):GOTO250
220 IFA$=CHR$(13)THENF=0:GOTO310
230 IF(A>31ANDA<95)OR(A>159ANDA<233)THEN
    250
240 PRINT:PRINT"{SH C}{SH A}{SH R}{SH A}
    {2 SH T}{SH E}{SH R}{SH E} {SH N}{SH
     O}{SH N} {SH C}{SH O}{SH N}{SH S}{S
    H E}{SH N}{SH T}{SH I}{SH T}{SH O}!"
    :PRINT:GOTO180
250 PRINTA$;
260 IFF=1THENB$=B$+A$:LS=LS+1:IFLEN(B$)=
    L(W)THENRETURN
270 F=1:GOTO190
```

```
280 C1=PEEK(205)+1:C2=PEEK(202):L=3072+4
    0*(C1-1)+C2:IFL<3072ORL>4071THENRETU
    RN
290 POKEL-1024,127:POKEL,160:FORI=1TO80:
    NEXT:POKEL,32:FORI=1TO80:NEXT:RETURN
300 REM **AGGIUNGE SPAZI SHIFTATI FINO A
     LUNGHEZZA CAMPO**
310 A=LEN(B$):D=L(W)-A:IFA=0THENRETURN
320 IFD>0THEN340
330 IFD<0THENB$=LEFT$(B$,L(W)):RETURN
340 A$=CHR$(160):C$="":FORI=1TOD:C$=C$+A
    $:NEXT:B$=B$+C$:RETURN
350 FORI=1TO3000:NEXT:RETURN
360 REM **DEFINIZIONE DEL FILE**
370 CLR:INPUT"{CLR}{SH Q}UANTI CAMPI PER
     RECORD";C:PRINT:TRAP850:DIMN$(C),L(
    C)
380 RL=0:FORI=1TOC
390 PRINT"{RVS ON}{SH C}AMPO N.{RVS OFF}
    "I:PRINT
400 INPUT"{SH N}OME";N$(I)
410 INPUT"{SH L}UNGHEZZA";L(I)
420 RL=RL+L(I):PRINT:NEXT
430 RL=RL+1:PRINT"{SH L}UNGHEZZA TOTALE
    DI UN RECORD:":PRINTRL"CARATTERI.":P
    RINT
440 IFRL>255THENPRINT"{SH R}{SH E}{SH C}
    {SH O}{SH R}{SH D} {SH T}{SH R}{SH O
    }{2 SH P}{SH O} {SH L}{SH U}{SH N}{S
    H G}{SH H}{SH I}!":FORI=1TO5000:NEXT
    :GOTO1180
450 RN=0:PRINT"{SH D}IMENSIONI DEL FILE
    ({SH N}.RECORD)";:INPUTRN:PRINT
460 TRAP850:DIMRC$(RN)
470 INPUT"{SH N}OME DEL FILE";F$:IFLEN(F
    $)>0ANDLEN(F$)<11THENF$=CHR$(18)+F$:
    GOTO1180
480 PRINT"{SH D}{SH A} 1 {SH A} 10 {SH C
```

```
    }{SH A}{SH R}{SH A}{2 SH T}{SH E}{SH
     R}{SH I}!":GOTO470
490 GETA$:IFA$=""THEN490
500 A=ASC(A$+CHR$(0)):RETURN
510 REM **INSERIMENTO DI UN RECORD**
520 R=TY+1
530 PRINT"{CLR}{RVS ON}+ {SH S}{SH U}{2
    SH C}. ¬ {SH P}{SH R}{SH E}{SH C}. \
     {SH S}{SH I}{SH N}{SH G}. {SH E} {S
    H E}{SH S}{SH C}{SH E}{2 SPC}{SH I}
    {SH I}{SH N}{SH S}{SH E}{SH R}."
540 PRINT"{SH R}{SH E}{SH C}{SH O}{SH R}
    {SH D} #"R
550 PRINT"{CUR.GIU}{SH S}CELTA?"
560 GOSUB490:IFA<>43ANDA<>45ANDA<>92ANDA
    <>69ANDA<>73THEN560
570 IFA=43THENR=R+1
580 IFA=43ANDR>TY+1THENPRINT"{SH E}SISTO
    NO RECORD VUOTI PIU'BASSI!":GOTO550
590 IFA=43THEN530
600 IFA=45THENR=R¬1:IFR<1THENR=1
610 IFA=45THEN530
620 IFA=92THENINPUT"{SH N}UMERO DEL RECO
    RD";R
630 IFA=92ANDR>TY+1THENPRINT"{SH E}SISTO
    NO RECORD VUOTI PIU'BASSI!":GOTO620
640 IFA=92THEN530
650 IFA=69THEN1180
660 RC$(R)="":FORW=1TOC:PRINT:B$=""
670 PRINTN$(W)"("L(W)"CARATT.):":GOSUB18
    0
680 RC$(R)=RC$(R)+B$
690 NEXT:IFR<=TYTHEN520
700 TY=TY+1:GOTO520
710 REM **LEGGE IL FILE**
720 R=1
730 GOSUB940:PRINT"{RVS ON}+ {SH S}{SH U
    }{2 SH C}.{3 SPC}¬ {SH P}{SH R}{SH E
    }{SH C}.{4 SPC}\ {SH S}{SH I}{SH N}{
```

```
    SH G}.{3 SPC}{SH E} {SH E}{SH S}{SH
    C}{SH E}{3 SPC}"
740 PRINT"{SH R}{SH E}{SH C}{SH O}{SH R}
    {SH D} #";R:PRINT
750 IFRC$(R)=""THENPRINT"{SH R}{SH E}{SH
     C}{SH O}{SH R}{SH D} {SH V}{SH U}{S
    H O}{SH T}{SH O}!":GOTO780
760 PS=1:FORK=1TOC:PRINTN$(K)":"MID$(RC$
    (R),PS,L(K)):PRINT:PS=PS+L(K)
770 NEXT:PRINT
780 PRINT"{CUR.GIU}{SH S}CELTA?":GOSUB49
    0
790 IFA<>43ANDA<>45ANDA<>92ANDA<>69THENG
    OSUB490:GOTO790
800 IFA=43THENR=R+1:GOTO730
810 IFA=45THENR=R-1:IFR<1THENR=1
820 IFA=45THEN730
830 IFA=92THENINPUT"{SH N}. RECORD";R:GO
    TO730
840 IFA=69THEN1180
850 REM **INTERCETTAZIONE ERRORI**
860 PRINT"{RVS ON}{SH A}{2 SH T}{SH E}{S
    H N}{SH Z}{SH I}{SH O}{SH N}{SH E}!
    {SH S}I E' VERIFICATO UN ERRORE!"
870 PRINT"{CUR.GIU}{SH E}RRORE:"ER:PRINT
    "{CUR.GIU}{SH T}IPO: "ERR$(ER)
880 IFER<>16THEN920
890 PRINT"{CUR.GIU}{SH O}CCORRE DIMINUIR
    E IL NUMERO DEI RECORD"
900 PRINT"OPPURE IL FILE E' TROPPO GRAND
    E PER"
910 PRINT"ESSERE GESTITO DA QUESTO COMPU
    TER."
920 GOSUB490:RESUME1650
930 REM **DATI DEL FILE**
940 COLOR1,4:PRINT"{CLR}{RED}{SH F}{SH I
    }{SH L}{SH E}:"F$;TAB(20)"{RVS OFF}{
    SH C}APACITA':"RN
```

```
950 PRINT"{SH C}AMPI:"C;TAB(20)"{SH R}EC
    ORD:"TY
960 PRINT"{SH A}GGIORN.AL:"U$TAB(20)"{SH
     O}RE:"LEFT$(T$,2)" "MID$(T$,3,2)" "
    RIGHT$(T$,2):COLOR1,2
970 PRINT"*******************************
    **********":RETURN
980 REM **SALVA IL FILE**
990 COLOR4,4:GOSUB940
1000 PRINT"{SH I}NSERISCI LA CASSETTA E
     PREMI {SH R}{SH E}{SH T}{SH U}{SH R
     }{SH N}"
1010 GOSUB490:IFA<>13THEN1010
1020 D$=STR$(PEEK(1630))+STR$(PEEK(1631)
     )+STR$(PEEK(1632))
1030 R$=CHR$(13):OPEN1,1,1,F$
1040 PRINT#1,RN;R$;TY;R$;C;R$;RL;R$;D$;R
     $;TI$
1050 FORI=1TOC:PRINT#1,N$(I);R$;L(I):NEX
     T
1060 FORI=1TOTY:PRINT#1,RC$(I):NEXT
1070 CLOSE1:COLOR4,1:GOTO1180
1080 REM **CARICA IL FILE**
1090 CLR:COLOR4,5:INPUT"{CLR} {SH N}OME
     DEL FILE";F$:F$=CHR$(18)+F$
1100 OPEN1,1,0,F$
1110 INPUT#1,RN,TY,C,RL,U$,T$:TRAP850:DI
     MN$(C),L(C),RC$(RN)
1120 FORI=1TOC:INPUT#1,N$(I),L(I):NEXT
1130 FORI=1TOTY:RC$(I)=""
1140 GET#1,A$:RC$(I)=RC$(I)+A$:IFA$<>CHR
     $(13)THEN1140
1150 NEXT
1160 CLOSE1:COLOR4,1:GOTO1180
1170 REM **MENU DI SELEZIONE**
1180 PRINT"{CLR}{RVS ON}{SH S}{SH E}{SH
     Q}{SH F}{SH I}{SH L}{SH I}{SH N}{SH
      G} 1.0{20 SPC}{SH M}{SH C}{SH G} 8
     5"
```

```
1190 A$="{5 CUR.DES}"
1200 PRINT"{5 CUR.GIU}"A$"0. {SH C}REA N
     UOVO FILE"
1210 PRINTA$"1. {SH C}ARICA FILE ESISTEN
     TE"
1220 PRINTA$"2. {SH S}ALVA FILE IN MEMOR
     IA"
1230 PRINTA$"3. {SH S}CRIVE UN RECORD"
1240 PRINTA$"4. {SH L}EGGE UN RECORD"
1250 PRINTA$"5. {SH R}ICERCA UN RECORD"
1260 PRINTA$"6. {SH O}RDINA ALFABETICAME
     NTE"
1270 PRINTA$"7. {SH M}EMORIA LIBERA"
1280 PRINTA$"8. {SH S}TAMPA IL FILE"
1290 PRINTA$"9. {SH F}INE OPERAZIONI"
1300 PRINT"{2 CUR.GIU}{SH S}CELTA?"
1310 GOSUB490:IFA<48ORA>57THEN1310
1320 ONA-47GOTO370,1090,990,520,720,1340
     ,1510,1670,1600,1700
1330 REM **RICERCA UN RECORD**
1340 PRINT"{CLR}{SH I} CAMPI SONO:":FORI
     =1TOC:PRINT"{SH C}AMPO {SH N}."I":"
     ,N$(I):NEXT
1350 PRINT"{CUR.GIU}{SH N}UMERO DEL CAMP
     O SU CUI EFFETTUARE LA"
1360 INPUT"RICERCA";A:IFA<1ORA>CTHEN1350
1370 PS=1:IFA>1THENFORI=1TOA-1:PS=PS+L(I
     ):NEXT
1380 INPUT"{CUR.GIU}{SH C}ONTENUTO DEL C
     AMPO";C$
1390 INPUT"{SH V}UOI UNO STAMPATO{2 SPC}
     N{3 CUR.SIN}";S$:IFS$="S"THENOPEN4,
     4,7:CMD4
1400 FORI=1TOTY:IFC$<>MID$(RC$(I),PS,LEN
     (C$))THEN1470
1410 PRINT"{CLR}{SH R}ECORD {SH N}."I
1420 PO=1:FORW=1TOC:IFPO=1THENPRINTN$(1)
     ":"LEFT$(RC$(I),L(1)):GOTO1440
1430 PRINTN$(W)":"MID$(RC$(I),PO,L(W))
```

```
1440 PO=PO+L(W):NEXT
1450 INPUT"{CUR.GIU}{SH P}ROSEGUO NELLA
     RICERCA (S/N){2 SPC}S{3 CUR.SIN}";A
     $
1460 IFA$<>"S"THEN1180
1470 NEXT
1480 IFS$="S"THENPRINT#4:CLOSE4
1490 PRINT"{CUR.GIU}{SH E}SITO NEGATIVO
     RICERCA. {SH P}REMI UN TASTO":GOSUB
     490:GOTO1180
1500 REM **ORDINAMENTO RECORD**
1510 PRINT"{CLR}{SH O}RDINAMENTO IN CORS
     O. {SH A}TTENDERE."
1520 F=0
1530 FORI=1TOTY-1:IFRC$(I)<=RC$(I+1)THEN
     1560
1540 A$=RC$(I):RC$(I)=RC$(I+1):RC$(I+1)=
     A$
1550 F=1
1560 NEXT
1570 IFF=1THEN1520
1580 GOTO1180
1590 REM **STAMPA IL FILE**
1600 PRINT"{CLR}{SH P}OSIZIONA LA CARTA
     SULLA STAMPANTE E{4 SPC}PREMI UN TA
     STO"
1610 GOSUB490:OPEN4,4,7:CMD4:FORR=1TOTY
1620 PS=1:FORK=1TOC:PRINTN$(K)":"MID$(RC
     $(R),PS,L(K)):PRINT:PS=PS+L(K)
1630 NEXT:PRINT:NEXT:PRINT#4:CLOSE4:GOTO
     1180
1640 REM **LINEA DI RESUME PER ROUTINE I
     NTERCETTAZIONE ERRORI**
1650 CLR:RUN1180
1660 REM **MEMORIA LIBERA**
1670 PRINT"{CLR}{SH A}TTENDERE...":BY=FR
     E(0)-(FRE(0)<0)*65536
1680 PRINT"{CLR}"BY"BYTE LIBERI":GOSUB35
     0:GOTO1180
```

```
1690 REM **FINE OPERAZIONI**
1700 INPUT"{CLR}{SH C}ONFERMA{2 SPC}N{3
     CUR.SIN}";A$:IFA$<>"S"THEN1180
1710 SYS32768
```

CAPITOLO 3

IL FOGLIO ELETTRONICO E LE OPERAZIONI SU DATI NUMERICI

3.1 La contabilità effettuata con il computer

Quasi tutti sanno esattamente che cosa sia un word processor od un database, mentre il concetto di 'foglio elettronico' rimane per lo più vago e del tutto sconosciuto.

In realtà, la tripletta di programmi di word processor, database e *spreadsheet* — nome inglese universalmente usato per i fogli elettronici — rappresenta quanto di meglio si possa utilizzare per lavorare con il proprio computer. Lo spreadsheet altro non è che la versione informatica di un normale foglio di carta, usato in contabilità per incolonnare cifre e numeri.

Uno spreadsheet è composto da una serie di *celle*, ognuna delle quali contiene un numero. Le celle sono individuate, con il principio delle coordinate cartesiane, dal loro numero di riga e numero di colonna. Il numero contenuto nella cella (3,10) sarà quindi il decimo valore della riga 3 o, in modo equivalente, il terzo valore della colonna 10. Ciò equivale a dire che il numero contenuto nella cella (3,10) si troverà in corrispondenza dell'intersezione tra la riga 3 e la colonna 10.

Per spiegarci meglio, ricorriamo a titolo di esempio ad un 'micro-spreadsheet', composto da 4 righe per 6 colonne. Esso potrà avere il seguente aspetto:

	Col.1	Col.2	Col.3	Col.4	Col.5	Col.6
Riga 1:	100	55	0	0	0	0
Riga 2:	250	125	0	0	0	0
Riga 3:	50	350	0	0	0	0
Riga 4:	500	200	0	0	0	0
Riga 5:	0	0	0	0	0	0

Nell'esempio, i dati sono stati inseriti soltanto nelle colonne 1 e 2; la cella (1,1) contiene il valore 100, la (1,2) il valore 55, la (2,1) il valore 250, e così via...
Rispetto al tradizionale foglio di carta per contabilità, uno spreadsheet offre molteplici vantaggi: tra questi possiamo elencare:

— Totale precisione nei calcoli: il rischio di commettere errori in calcoli lunghi e ripetitivi viene completamente eliminato dall'infallibilità del computer in queste operazioni.

— Estrema facilità di modifiche all'interno del foglio: per modificare un numero in una cella, basta posizionarvisi sopra con il cursore e ridigitarlo secondo le proprie esigenze.

— Riutilizzabilità: una volta definite le formule di calcolo che ci occorrono, è semplicissimo compiere delle analisi cosiddette *what if?* (cosa succede se?...) modificando i parametri e ricalcolando i totali.

— Ottima leggibilità dei risultati: se si possiede una stampante, il contenuto dello spreadsheet può: essere riversato su carta in forma di tabulato, con la implicita leggibilità.

Uno spreadsheet può essere utilizzato per i più molteplici scopi, come il calcolo di budget, previsioni di investimento, calcolo di profitti e perdite, resoconti di spese sostenute, e comunque per tutti quei compiti che implicano l'uso intensivo di dati numerici.
Tutti i 'pacchetti software integrati' per i principali personal computer di classe superiore (dai 128 Kbyte in su, per intenderci) offrono esattamente i tre programmi che trovate in questo libro: uno spreadsheet per tenere la contabilità, un database per gestire l'archivio, ed un word processor per generare la corrispondenza ed i rapporti aziendali. Naturalmente il C16 non è un computer adatto ad impieghi eminentemente gestionali, tuttavia, se ne accettiamo le limitazioni soprattutto in termini di memoria disponibile e velocità di esecuzione, con i tre programmi qui presentati saremo perfettamente in grado di utilizzarlo per le nostre esigenze.

3.2 Multicalc V1: uno spreadsheet in 12 Kbyte

Il programma consiste in uno spreadsheet che occupa — quando è pieno di dati — praticamente tutta la memoria disponibile nel C16. La forma grafica nella quale si presenta è simile a quella dello spreadsheet del 'fratello maggiore' Plus / 4 al quale ci si è ispirati anche nel tipo di inserimento dei comandi.
Le celle disponibili sono 360, organizzate in 40 righe e 9 colonne. Dal momento che non è possibile, per le insufficienti dimensioni dello schermo, avere visualizzato tutto lo spreadsheet in una sola volta, esso è stato suddiviso in sei *pagine* da

venti righe e tre colonne ciascuna. Nella parte alta dello schermo sono visualizzati i numeri di colonna, mentre nella parte sinistra i numeri di riga della pagina corrente; il grosso cursore rettangolare definisce la cella sulla quale si sta attualmente lavorando.
La parte bassa dello schermo costituisce l'*area di comando*; in essa si trova l'indicatore della riga e colonna corrente (cioè della cella corrente sulla quale si trova il cursore), e la linea di inserimento dei comandi preceduta da una 'C'. Una scritta al centro dell'area di comando visualizza il tipo di memoria di massa si cui è regolato correntemente il programma.
MULTICALC è dotato di una serie di 18 comandi per l'editing e per l'effettuazione dei calcoli, oltre che di tre diversi modi di operazione. Vediamo ora, nell'ordine, quali essi siano.
Una volta attivato il programma, viene visualizzata la prima pagina dello spreadsheet (righe da 1 a 20 e colonne da 1 a 3), con il cursore lampeggiante in corrispondenza della cella (1,1). Dal momento che lo spreadsheet è vuoto, dovremo inserire qualcosa nelle celle: abbiamo la possibilità di inserirvi sia del testo che dei numeri.
I dati vanno inseriti una cella alla volta, e nella cella sulla quale si trova il cursore. Poniamo il caso di voler inserire il testo "ricavi" nella cella (3,2): con i tasti cursore (quelli contrassegnati dalle frecce) ci porteremo sopra la cella (3,2). Verificheremo quindi che la cella sulla quale ci troviamo sia proprio quella desiderata, dando uno sguardo all'indicatore di cella nella parte in basso a sinistra dello schermo.
Potremo ora premere in tasto 'C' per entrare nel *modo comando*: il fatto che ci troviamo in modo comando viene evidenziato da una scritta lampeggiante in basso a destra. In modo comando, abbiamo quattro diverse possibilità di scelta:

1. Dare effettivamente un comando allo spreadsheet.
2. Entrare, con la pressione contemporanea dei tasti 'Commodore' e 'T', in modo inserimento testi.
3. Entrare, con la pressione contemporanea dei tasti 'Commodore' e 'N', in modo inserimento numeri.
4. Entrare, con la pressione contemporanea dei tasti 'Commodore' e 'F', in modo inserimento formule.

Dal momento che desideriamo inserire del testo nella cella, premeremo 'Commodore' e 'T': il fatto che ci troviamo ora in *modo testo* viene evidenziato da un scritta in basso a destra.
Abbiamo a disposizione dieci caratteri per ogni cella: se il testo da inserire — come nel nostro caso della stringa "ricavi" — è più corto di dieci caratteri, dovremo premere RETURN a fine testo; se il testo al contrario arriva all'undicesimo carattere, esso viene automaticamente inserito in cella senza che sia necessaria la pressione di RETURN.
Digitate quindi "ricavi" e premete RETURN. Comparirà la scritta ATTENDERE, ed il programma aggiornerà la videata con la cella (3,2) contenente la stringa

"ricavi". Il cursore lampeggiante si troverà ancora sulla cella (3,2). Possiamo ora spostarci su un'altra cella e ripetere l'operazione con un testo diverso, oppure rimanere sulla stessa cella e modificare, con procedimento analogo, la stringa "ricavi" in "costi".
È consigliabile effettuare un po' di esperimenti con il modo testo inserendo testi diversi in celle qualsiasi.
Si noti a questo punto che, se ci si trova su una cella della colonna 3 e si preme il tasto di spostamento a destra del cursore, compare la scritta ATTENDERE e lo spreadsheet visualizza la pagina seguente (colonne 4, 5 e 6) portandosi con il cursore sulla colonna 4. Analogamente, se ci si trova su una cella della riga 20 e si preme il tasto di spostamento verso il basso del cursore, lo spreadsheet si aggiorna sulla pagina composta dalle righe da 21 a 40.
Dopo aver imparato come inserire dei testi, vediamo come inserire dei valori numerici.
Portiamoci sulla cella desiderata e premiamo 'C' per entrare in modo comando; premiamo quindi contemporaneamente 'Commodore' e 'N' per comunicare al computer che intendiamo inserire un numero. Il computer visualizzaerà una scritta in basso a destra, per ricordarci che ci troviamo in *modo numerico.*
Possiamo ora digitare il numero desiderato, eventualmente preceduto dal segno '–' se negativo. Si usi ovviamente il punto e non la virgola per separare la parte intera da quella decimale. Come per il modo testo, il massimo numero di caratteri inseribili è dieci. Un numero potrà quindi essere composto da un massimo di 10 cifre se intero positivo, di 9 se intero negativo o se decimale positivo, e di 8 cifre se decimale negativo.

Alla pressione di RETURN o al raggiungimento dell'undicesimo carattere il numero viene inserito nella cella corrente. La pressione di tasti non numerici — con l'esclusione di '–', '.', '/' — viene rifiutata dal computer. Si noti che un numero può essere inserito in cella anche se ci si trova in modo testo: il modo numerico ha unicamente la funzione di evitare la pressione involontaria di tasti alfabetici mentre si desiderano inserire esclusivamente dei numeri: se l'inserimento di un numero avviene correttamente, non esiste alcuna differenza nel risultato tra il modo testo e il modo numerico.
Il cursore lampeggiante si troverà ancora sulla cella corrente. Potete ora spostarvi su un'altra cella e ripetere l'operazione con un altro numero, oppure rimanere sulla stessa cella e modificare, con procedimento analogo, il numero in essa contenuto.

È consigliabile effettuare un po' di esperimenti con il modo numerico, inserendo numeri in celle qualsiasi.
Dopo esservi impratichiti un po' con il modo testo ed il modo numerico, potrete scoprire il *modo formule.* Supponiamo si desideri sommare il contenuto della cella (1,1) al contenuto della cella (1,2), ed inserire il risultato nella cella (1,3). A titolo di esempio, inserite nella cella (1,1) il numero 110, e nella cella (1,2) il numero 90.

Dopo aver inserito questi due numeri, riposizionatevi con il cursore sulla cella (1,1), premete nuovamente 'C' per entrare in modo comando, quindi premete contemporaneamente 'Commodore' e 'F' per entrare in modo formule. L'ingresso in modo formule viene evidenziato da una scritta in basso a destra.
Premete il tasto '+'. Immediatamente sopra la linea di comando verrà visulizzato in campo inverso:

01,01+

Portatevi con il cursore sulla cella (1,2), premete 'C', premete 'Commodore' e 'F', premete '='. Sopra la linea di comando comparirà:

01,01+01,02=

Poiché vogliamo inserire il risultato nella cella (1,3), ci porteremo sopra di essa con il cursore, entreremo come prima in modo formule, e digiteremo '@' per comunicare al computer la fine della formula. Sopra la linea di comando comparirà:

01,01+01,02=01,03

e la formula verrà eseguita: lo spreasheet si aggiornerà visualizzando il risultato nella cella (1,3).
Come abbiamo visto, il sistema di inserimento delle formule è molto semplice e segue gli schemi dell'aritmetica elementare; gli operatori consentiti sono i consueti '+', '–', '*', '/', con l'aggiunta di '@' per segnalare la fine della formula. Ritorneremo sulle formule nel prossimo paragrafo, quando andremo ad esaminare il file dimostrativo contenuto sulla cassetta allegata al volume.
Passiamo ora all'analisi dei comandi disponibili all'interno dello spreadsheet: a tutti i comandi si accede entrando in modo comando con la pressione di 'C', digitando il comando di due caratteri seguito dagli eventuali parametri richiesti, e premendo RETURN. Non devono mai essere inseriti spazi tra un carattere e l'altro. Se il comando non ha la forma corretta, appare in basso a destra il messaggio SYNTAX ERROR accompagnato dal suono di un cicalino, ed il controllo ritorna al cursore.
Ecco i comandi disponibili in MULTICALC:

HM Porta il cursore in posizione di HOME, corrispondente alla cella (1,1).

GOx,y Porta il cursore sulla cella contraddistinta dalla riga x e dalla colonna y. Se ad x ed y vengono assegnati valori superiori rispettivamente e 49 e 9, il cursore si porta in ogni caso sull'ultima cella dello spreadsheet, la (40,9).

CRx,y Copia il contenuto della riga x nella riga y. Dopo questo comando, la riga y sarà esattamente uguale alla riga x.

CCx,y Copia il contenuto della colonna x nella colonna y. Dopo questo comando, la colonna y sarà esattamente uguale alla colonna x.

ZR Azzera completamente lo spreadsheet, cancellando il contenuto di tutte le celle. Esso non potrà più essere ricuperato, a meno che non sia precedentemente provveduto ad un suo salvataggio su memoria di massa.

S:nomefile Salva sulla memoria di massa il contenuto di tutte le celle. Il nome del file ha una lunghezza massima di otto caratteri.

L:nomefile Carica il contenuto dello spreadsheet dalla memoria di massa. Il nome del file ha una lunghezza massima di otto caratteri. Se si utilizza come memoria di massa il registratore a cassette, è possibile omettere il nome del file oppure digitarne soltanto i primi caratteri. Se si utilizza il disk drive, è possibile utilizzare i caratteri a 'wild card' come l'asterisco ed il punto interrogativo (consultare il manuale del proprio disk drive).

CPxynm Copia il contenuto di una cella in un'altra cella. I numeri di riga e colonna di entrambe le celle devono essere costituiti da due cifre e non separati da virgole. Ad esempio, per copiare il contenuto della cella (1,2) nella cella (32,9) si userà la sintassi: cp01023209. Dal momento che la sintassi richiesta occupa sempre 10 caratteri, non è necessario premere RETURN al termine del comando.

VF Visualizza sullo schermo le formule che sono state precedentemente definite. L'utente dispone della possibilità di definire cinque diverse formule.

CFx Cancella la formula x. Utile nel caso si abbia commesso qualche errore nell'inserire la formula. Esempio: cf2 cancella la formula numero 2.

FRx Esegue la formula numero x per tutte le righe dello spreadsheet. La formula deve ovviamente essere definita per calcoli sulla stessa riga.

FCx Esegue la formula numero x per tutte le colonne dello spreadsheet. La formula deve ovviamente essere definita per calcoli sulla stessa colonna.

ME Visualizza l'ammontare di memoria libera disponibile, in basso a destra sullo schermo.

SCx Esegue la somma di tutti i valori numerici contenuti nella colonna x, e ne visualizza il risultato nell'ultima riga.

SRx Esegue la somma di tutti i valori numerici contenuti nella riga x, e ne visualizza il risultato nell'ultima colonna.

SS Stampa lo spreadsheet. La stampa avviene in tre passate successive di pagine composte da 40 righe e 3 colonne ciscuna.

SFnomefile Salva sulla memoria di massa le formule precedentemente definite. Il nome del file ha una lunghezza massima di otto caratteri.

LFnomefile Carica delle formule precedentemente definite dalla memoria di massa. Il nome del file ha una lunghezza massima di otto caratteri. Se si utilizza come memoria di massa il registratore a cassette, è possibile omettere il nome del file oppure digitarne soltanto i primi caratteri. Se si utilizza il disk drive, è possibile utilizzare i caratteri a 'wild card' come l'asterisco ed il punto interrogativo (consultare il manuale del propro disk drive).

Infine, un cenno sul tipo di memoria di massa utilizzabile: quando ci si trova in modo cursore (cioè il modo normale che è attivato quando si dà il RUN al programma), premendo il tasto 'P' (per Periferica) si può passare dalla regolazione del programma su disk drive a quella su registratore a cassette, e viceversa. Il tipo di memoria di massa corrente è visualizzato nella parte bassa dello schermo.

3.3 Un esempio di utilizzo di Multicalc V1

Dal momento che, quasi sempre, la prima volta che si ha a che fare con uno spreadsheet ci si trova un po' disorientati, abbiamo inserito sulla cassetta allegata al volume un file dimostrativo. Esso si trova immediatamente dopo il programma "MULTICALC", e consiste in un esempio di operazioni su dati relativi alle vendite di un certo numero di prodotti.
Utilizzeremo questo file per mostrare come opera MULTICALC. Occorre innanzi tutto caricare quest'ultimo, premere a fine caricamento il tasto STOP sul registratore, e dare il RUN.
Per caricare il file dimostrativo, premete 'P' per regolare MULTICALC sull'unità a cassette, quindi premete 'C' per entrare in modo comando, e digitate:

l:demo

Assicuratevi di non avere inserito spazi tra i caratteri (gli errori di battitura possono essere corretti, come di consueto, con il tasto DEL), e premete RETURN. Comparirà la consueta scritta PRESS PLAY ON TAPE nella parte bassa dello schermo: premete il tasto PLAY ed attendete. Lo schermo scomparirà, ricomparendo a tratti durante il caricamento del file; non premete alcun tasto né sulla tastiera né sul registratore fino a che non venga visualizzato lo spreadsheet pieno di numeri. Solo a questo punto potete premere il tasto STOP sul registratore.
Avete ora in memoria il file dimostrativo: se desiderate, per praticità, salvarlo su

disco, premete 'P' per regolare MULTICALC sull'unità a dischi, entrate in modo comando e digitate:

s:demo

facendolo seguire da RETURN.
Vediamo ora che cosa rappresentano questi numeri all'interno dello spreadsheet: ogni riga contiene i dati relativi ad un certo articolo venduto, chiamato qui generalmente 'articolo'. Le colonne contengono invece i seguenti parametri:

colonna 1: nome dell'articolo.
colonna 2: costo di produzione unitario lordo.
colonna 3: prezzo di vendita unitario lordo.
colonna 4: ammontare dell'I.V.A. sul prezzo di vendita, diviso per 100.
colonna 5: numero di articoli venduti.

Le colonne dalla 6 alla 9 non contengono dati, e verranno utilizzate per i risultati dei calcoli.
Supponiamo ora di voler calcolare:

1. Costo totale lordo di produzione per ogni articolo.
2. Guadagno totale netto sul venduto per ogni articolo e guadagno globale.
3. Ammontare totale dell'I.V.A. sul venduto per ogni articolo.
4. Guadagno netto per ogni unità venduta, e per ogni articolo.

A questo scopo, organizzeremo le colonne dello spreadsheet in questo modo:

colonna 6: costo totale lordo di produzione per ogni articolo.
colonna 7: guadagno totale netto sul venduto per ogni articolo.
colonna 8: ammontare totale dell'I.V.A. sul venduto per ogni articolo.
colonna 9; guadagno netto per ogni unità venduta, e per ogni articolo.

Per effettuare i calcoli necessari ad ottenere questi risultati, dovremo introdurre in MULTICALC le opportune formule. Iniziamo dalla prima operazione, il calcolo del costo totale lordo di produzione. Dal momento che disponiamo, come dati di partenza, del costo unitario e del numero di pezzi prodotti, esso sarà dato dal loro prodotto.
Il costo unitario di produzione si trova sulla colonna 2, mentre il numero di pezzi prodotti sulla colonna 5. Il risultato andrà inserito nella colonna 6.
Posizionatevi con il cursore sulla cella (1,2), entrate in modo formule premendo 'C' seguito da 'F', e premete '*'. Il computer visualizzerà, sopra la linea di comando, la scritta in campo inverso:

01,02*

Portatevi ora con il cursore sopra la cella (1,5), che si trova nella pagina seguente verso destra; sarà necessaria una breve attesa per il cambio di pagina dello spreadsheet. Entrate come prima in modo formule e premete '='. Il computer visualizzerà:

01,02*01,05=

Poiché il risultato dell'operazione dovrà essere inserito nella colonna 6, portatevi con il cursore sulla cella (1,6), entrate in modo formule e premete '@' per indicare al computer la fine della formula. Verrà visualizzato:

01,02*01,05=01,06

Questa è infatti la formula occorrente per il calcolo del costo totale lordo di produzione dell'articolo numero 1: costo unitario (1,2) moltiplicato per numero pezzi (1,5) uguale a costo totale (1,6). A questo punto il computer avrà già inserito nella cella (1,6) il risultato, che sarà pari a 14300. Verificatelo portandovi con il cursore su questa cella.
Naturalmente ci occorre questo risultato per tutti quanti gli articoli contenuti nello spreadsheet, e non soltanto per l'articolo numero 1. Ormai abbiamo inserito la formula necessaria (è naturalmente ancora nella memoria del C16: verificatelo entrando in modo comando e digitando 'vf' seguito da RETURN per visualizzare le formule in memoria) e sarà sufficiente un solo comando per avere istantaneamente il costo totale lordo per ogni articolo.
A questo scopo, entrate in modo comando e digitare 'fr1' seguito da RETURN. Questo comando ordina al computer di eseguire i calcoli della formula numero 1 su tutte le righe dello spreadsheet. Quando sarà scomparsa la scritta lampeggiante "IN CORSO", potrete portarvi con il cursore su una cella qualunque della colonna 6 per verificare che vi siano riportati tutti i totali.
Sappiamo ora a quanto ammontano i costi globali di produzione per ogni articolo. Il secondo risultato che ci occorre è l'ammontare dell'IVA sul venduto. Esso potrà essere calcolato come il prodotto del prezzo lordo unitario per coefficiente IVA per il numero di pezzi venduti. Facendo sempre riferimento alla riga 1, la formula da inserire sarà:

01,03*01,04*01,05=01,08

dal momento che intendiamo inserire il risultato nella colonna 8 ed i tre termini del prodotto si trovano rispettivamente nelle colonne 3, 4 e 5. Alla pressione del tasto '@' di fine formula, il risultato, pari a 27522, verrà inserito nella cella (1,8). Analogamente a quanto fatto prima, per estendere il calcolo a tutti gli altri articoli sarà sufficiente entrare in modo comando e digitare

fr2

Il numero che segue il comando 'fr' è questa volta 2, perché la formula che desideriamo utilizzare è la numero due.
Per calcolare il guadagno totale netto su ogni tipo di articolo, dovremo sottrarre dal prodotto tra prezzo di vendita lordo unitario e numero di pezzi venduti (cioè dal ricavo lordo) i due valori prima calcolati, cioè costo totale lordo di produzione ed ammontare totale dell'IVA. La formula di calcolo sarà dunque:

01,03*01,05-01,08-01,06=01,07

Il risultato di 111078 verrà così inserito nella cella (1,7). Analogamente a quanto fatto precedentemente, con il comando 'fr3' eseguiremo questo calcolo per tutti gli articoli.
L'ultimo valore che ci siamo prefissi di calcolare è il guadagno netto per ogni pezzo venduto di ogni articolo. Esso sarà dato dal guadagno totale netto diviso per il numero di pezzi venduti. La formula relativa sarà:

01,07/01,05=01,09

Il risultato di 2019.6 verrà immesso nella cella (1,9), e con il comando 'fr4' otterremo il valore relativo agli altri articoli.
A questo punto abbiamo riempito tutte le celle dello spreadsheet, ma non abbiamo esaurito i nostri calcoli. Dobbiamo infatti calcolare quanto abbiamo guadagnato in totale con le vendite di tutti gli articoli in catalogo. Poiché la colonna interessata è la 7, con il semplice comando:

sc7

il computer inserirà nell'ultima cella della colonna 7 il risultato della somma di tutti i valori contenuti nella colonna stessa. La cella (40,7) conterrà dunque il guadagno totale. Analogamente, con i comandi 'sc6' ed 'sc8' potremo avere i totali dei costi e dell'IVA.
Se desideriamo a questo punto una stampa dello spreadsheet, sarà sufficiente dare il comando 'ss'.
Abbiamo ora in memoria tutte e quattro le formule necessarie a questi calcoli, e possiamo effettuare la prima analisi 'what if?'. Supponiamo di voler vedere di quanto varierebbero i nostri guadagni aumentando la produzione dell'articolo numero 3 da 94 a 150 unità, e diminuendo contemporaneamente quella dell'articolo numero 4 da 98 a 50 unità.
Posizioniamoci con il cursore sulla cella (3,5), premiamo 'C' seguito da 'N' per entrare in modo numerico, e digitiamo 150. Il nuovo valore viene inserito nella cella (3,5). Con procedura analoga, portiamoci sulla cella (4,5) e modifichiamone il contenuto in 50.
Sarà ora sufficiente dare, uno alla svolta, i comandi 'fr1', 'fr2', 'fr3', 'fr4' e 'sc7' per avere i nuovi valori relativi alla situazione mutata, che potranno essere

confrontati con il tabulato dei precedenti: in pochi istanti il computer ha svolto una serie di operazioni che, con carta e matita, avrebbe richiesto un tempo infinitamente superiore.
E che cosa succederebbe ('what if?') se aumentassimo il prezzo di un articolo e ne diminuissimo un altro? Non dovete fare altro che modificare i parametri che vi interessano, fare rieseguire i calcoli al computer, e leggere il risultato.
Naturalmente questo è soltanto un esempio banale delle possibili applicazioni di MULTICALC, che si adatteranno perfettamente alle vostre esigenze.
Un consiglio importante: nell'utilizzo delle formule, è molto importante il loro ordine di esecuzione. Alcune formule vanno utilizzate prima di altre, al fine di porre nella giusta casella un valore che sarà utilizzato dalle formule seguenti per proseguire nei calcoli. Se questo valore non è ancora stato calcolato, sarà molto facile tentare di far eseguire al computer una divisione per zero, ottenendo l'inevitabile messaggio di errore. Se le formule (che vanno sempre studiate a tavolino) non seguono il corretto ordine di esecuzione, i risultati saranno sicuramente falsati e quindi non veritieri.
Le formule in memoria possono essere salvate su nastro o disco, per utilizzi futuri, con il comando 'sfnomefile', e ricaricate con il comando 'lfnomefile'.

3.4 Commento al listato

10 linea DATA contenente la tabella dei comandi disponibili

20 annullamento delle funzioni predefinite dei tasti funzione ed inzializzazione routine intercettazione errori

30 subroutine visualizzazione messaggio durante l'effettuazione delle operazioni

40-50 inizializzazione generale. Se si utilizzano espansioni di memoria, lo spreadsheet può essere espanso come numero di righe o di colonne, aumentando rispettivamente il valore delle variabili MR ed MC. Il valore di MR deve sempre essere un multiplo di 20, mentre quello di MC deve essere multiplo di 3; i loro valori massimi, se modificati, dipendono dal tipo di espansione collegata e vanno trovati sperimentalmente

70-120 stampa su video della maschera dello spreadsheet, ed inizializzazioni delle variabili di controllo

140-210 subroutine che permette la visualizzazione di una nuova pagina dello spreadsheet, quando il cursore supera uno dei confini della pagina precedente

230-300 subroutine di posizionemento del cursore all'interno della pagina corrente dello spreadsheet

320-430 main loop del programma, in cui si attende la pressione di uno dei tasti cursore, oppure del tasto 'P' per modificare il tipo di memoria di massa (indicato dalla variabile DV), oppure del tasto 'C' per entrare in modo comando

450-580 loop secondario per il modo comando: il primo tasto premuto indica al computer cosa deve aspettarsi: se numeri, caratteri alfanumerici, formule o comandi

590-670 queste linee vengono eseguite se si è impartito un comando: la sua sintassi viene controllata e, in caso di errori, viene visualizzato il messaggio "Syntax error" accompagnato da un cicalino. Il controllo viene quindi passato alla subroutine relativa, in linea 650

690-710 vengono cancellate le scritte eventualmente apparse nella parte inferiore destra dello schermo

730-740 subroutine di cançellazione di una formula

760 subroutine di 'GOTO' che permette di spostarsi con il cursore direttamente sulla cella desiderata

780-820 subroutine di estrazione dei parametri relativi al comando assegnato, con ulteriore controllo sulla loro sintassi

849 subroutine di copia del contenuto di una certa riga all'interno di un'altra riga

880 subroutine di azzeramento del contenuto dello spreadsheet (ritorno alle condizioni iniziali)

900-930 subroutine di salvataggio dello spreadsheet sulla memoria di massa corrente

950-980 subroutine di caricamento dello spreadsheet dalla memoria di massa corrente

1000-1020 subroutine di copia del contenuto di una cella in un'altra cella

1040-1100 subroutine di controllo e generazione delle formule di calcolo, che vengono immagazzinate nel vettore di stringa S$()

1120-1160 subroutine di visualizzazione delle formule correntemente in memoria

1180-1300 esecuzione delle formule. Lo spreadsheet è regolato su risultati in virgola mobile. Se, per particolari applicazioni, fosse richiesto soltanto il valore

intero dei risultati, è sufficiente modificare il 'VAL(C$(B1,B2))' nelle linee 1250-1280 in 'INT(VAL(C$(B1,B2)))'. Questo in particolare perché, essendo il contenuto delle celle limitato a 10 caratteri, se la parte decimale di un numero fosse composta da troppe cifre, si potrebbe avere un troncamento della sua parte intera. Questo troncamento avverrebbe tuttavia soltanto nella visualizzazione dello spreadsheet, mentre il valore contenuto in memoria rimarrebbe comunque quello corretto

1320-1390 subroutine di calcolo di una formula per tutte le righe dello spreasheet

1410-1480 subroutine di calcolo di una formula per tutte le colonne dello spreadsheet

1500 subroutine di visualizzazione dell'ammontare di memoria libera disponibile correntemente

1520-1530 subroutine di esecuzione della somma di tutti i valori contenuti in una colonna

1550-1560 subroutine di stampa dello spreadsheet. Se si è modificato il valore della variabile MC e si dispone di un'espansione di memoria, l'estremo superiore del ciclo FOR...NEXT di linea 1580, attualmente 7, va modificato in (MC-2)

1640-1750 subroutine di intercettazione degli errori con rientro all'istruzione successiva a quella che ha generato l'errore. Se l'errore si verifica durante un ciclo FOR...NEXT, questa routine viene richiamata tante volte quante viene eseguito ciclo stesso

1770-1800 subroutine di salvataggio delle formule sulla memoria di massa corrente

1820-1870 subroutine di caricamento delle formule dalla memoria di massa corrente

MULTICALC V1

```
10 DATAHM,GO,CC,CR,ZR,"S:","L:",CP,VF,CF
   ,FR,FC,ME,SC,SR,SS,SF,LF
20 TRAP1640:FORI=1TO8:KEYI,"":NEXT:DV=8:
   GOTO40
```

```
30 CHAR1,25,22,"{14 SPC}":CHAR1,25,23,"{
   FLASH ON}{SH I}{SH N} {SH C}{SH O}{SH
    R}{SH S}{SH O}...{FLASH OFF}":RETURN
40 COLOR1,2:COLOR0,1:COLOR4,1:PRINTCHR$(
   14):SCNCLR:MR=40:MC=9:Z=1:VOL8
50 DIMC$(MR,MC),S$(5):GOSUB70:CHAR1,5,1,
   "{FLASH ON}{RVS ON}{SH M}{SH U}{SH L}
   {SH T}{SH I}{SH C}{SH A}{SH L}{SH C}!
   {FLASH OFF}{RVS OFF}":GOTO320
60 REM **MASCHERA**
70 PRINT"{RVS ON}BY {SH M}{SH C}{SH G}{3
    SPC}{SH C} 1{8 SPC}{SH C} 2{8 SPC}{S
   H C} 3{6 SPC}"
80 FORI=1TO20:A$=STR$(I):A=LEN(A$):A$=RI
   GHT$(A$,A¬1):IFA<3THENA$="0"+A$
90 PRINT"{RVS ON}{SH R} "A$:NEXT
100 PRINT"{40 CBM Y}":CHAR1,15,22,"DISCO
    "
110 CHAR1,0,24,"{SH C}> ¬¬¬¬¬¬¬¬¬¬¬":CHAR
    1,0,22,"{SH R} 1":CHAR1,5,22,"{SH C}
    {SH SPC}1"
120 R=1:C=1:F=0:X=5:Y=1:C1=1:R1=1:GOSUB2
    30:RETURN
130 REM **VISUALIZZAZIONE NUOVA PAGINA**
140 CHAR1,29,20,"{FLASH ON} {SH A}{2 SH
    T}{SH E}{SH N}{SH D}{SH E}{SH R}{SH
    E}{FLASH OFF}"
150 Q$="":CHAR1,10,0,"{RVS ON}{3 SPC}":C
    HAR1,21,0,"{RVS ON}{3 SPC}":CHAR1,32
    ,0,"{RVS ON}{3 SPC}":CHAR1,0,0,"{RVS
     ON}{6 SPC}"
160 CHAR1,10,0,"{RVS ON}"+STR$(C1):CHAR1
    ,21,0,"{RVS ON}"+STR$(C1+1):CHAR1,32
    ,0,"{RVS ON}"+STR$(C1+2)+"{RVS OFF}"
170 A=15:FORJ=C1TOC1+2:FORI=R1TOR1+19:K=
    I¬INT((I¬1)/20)*20
180 A$=STR$(I):L=LEN(A$):A$=RIGHT$(A$,L¬
    1):IFLEN(A$)<2THENA$="0"+A$
```

```
190 CHAR1,2,K,"{RVS ON}"+A$+"{RVS OFF}":
    CHAR1,A-10,K,"{10 SPC}"
200 CHAR1,A-10,K,"{10 SPC}":CHAR1,A-LEN(
    C$(I,J)),K,C$(I,J)
210 NEXT:A=A+12:NEXT:RETURN
220 REM **MOVIMENTO CURSORE**
230 CHAR1,X,Y,Q$:CHAR1,0,22,"{11 SPC}":C
    HAR1,0,22,"{SH R}"+STR$(R):CHAR1,5,2
    2,"{SH C}"+STR$(C)
240 IFC=INT(C/3)*3THENX=29:GOTO260
250 IFC-INT(C/3)*3=2THENX=17:ELSEX=5
260 Y=R-INT(R/20)*20:IFR/20=INT(R/20)THE
    NY=20
270 LS=3072+Y*40+X:Q$="":FORI=LSTOLS+9:K
    =PEEK(I)
280 IFK<27THENK=K+64
290 Q$=Q$+CHR$(K):NEXT
300 CHAR1,X,Y,"{FLASH ON}{RVS ON}"+Q$+"{
    FLASH OFF}{RVS OFF}":RETURN
310 REM **CICLO PRINCIPALE**
320 GETKEYA$
330 IFA$="C"THEN450
340 IFA$="{CUR.GIU}"ANDR=MRTHEN430
350 IFA$="{CUR.GIU}"THENR=R+1:IFY=20THEN
    R1=R:GOSUB140
360 IFA$="{CUR.SU}"ANDR=1THEN430
370 IFA$="{CUR.SU}"THENR=R-1:IFY=1THENR1
    =R-19:GOSUB140
380 IFA$="{CUR.DES}"ANDC=MCTHEN430
390 IFA$="{CUR.DES}"THENC=C+1:IFX=29THEN
    C1=C:GOSUB140
400 IFA$="{CUR.SIN}"ANDC=1THEN430
410 IFA$="{CUR.SIN}"THENC=C-1:IFX=5THENC
    1=C-2:GOSUB140
420 IFA$="P"THENDV=DV+7:CHAR1,15,22,"DIS
    CO ":IFDV>8THENDV=1:CHAR1,15,22,"NAS
    TRO"
430 GOSUB230:GOTO320
440 REM **MODO COMANDO**
```

```
450 B$="":I=3:F=0:SOUND1,800,10:CHAR1,25
    ,22,"{FLASH ON}{SH M}ODO COMANDO{FLA
    SH OFF}"
460 CHAR1,0,23,"{39 SPC}"
470 GETKEYA$
480 IFA$="{CBM T}"THENGOSUB690:CHAR1,25,
    23,"{SH M}ODO TESTO":F=1:GOTO470
490 IFA$="{CBM N}"THENGOSUB690:CHAR1,25,
    23,"{SH M}ODO NUMERICO":F=3:GOTO470
500 IFA$="{CBM F}"THENGOSUB690:CHAR1,25,
    23,"{SH M}ODO FORMULE":F=2:GOTO470
510 IFA$=CHR$(13)ANDF<>2THEN590
520 IFA$=CHR$(20)THENCHAR1,I-1,24," ":B$
    =LEFT$(B$,LEN(B$)-1):I=I-1:GOTO470
530 IFF=1ANDASC(A$)>128THEN470
540 IFF=3AND(ASC(A$)<45ORASC(A$)>57)THEN
    470
550 IFF=2ANDA$<>"+"ANDA$<>"-"ANDA$<>"*"A
    NDA$<>"/"ANDA$<>"="ANDA$<>"@"THEN640
560 IFF=2THEN600
570 CHAR1,I,24,A$:I=I+1:B$=B$+A$:IFLEN(B
    $)=10THEN590
580 GOTO470
590 IFF=1ORF=3THENC$(R,C)=B$:GOSUB140:GO
    TO660
600 IFF=2THENW$=A$:GOSUB1040:GOTO660
610 A$=LEFT$(B$,2):K=1:RESTORE
620 READC$:IFA$=C$THEN650
630 K=K+1:IFK<19THEN620
640 CHAR1,25,23,"{SH S}YNTAX ERROR":SOUN
    D1,200,10:FORK=1TO500:NEXT:GOTO660
650 ONKGOSUB710,760,840,860,880,900,950,
    1000,1120,730,1320,1410,1500,1520,15
    50,1580,1770,1820
660 CHAR1,0,24,"{SH C}> [illegible]":FORI
    =22TO23:CHAR1,25,I,"{14 SPC}":NEXT
670 GOTO370
680 REM **SUBR.CANCELLAZIONE FLAG MODO**
```

```
690 FORI=23TO23:CHAR1,25,I,"{14 SPC}":NE
    XT:I=3:RETURN
700 REM **HOME (HM)**
710 R=1:C=1:R1=R:C1=C:GOSUB140:RETURN
720 REM **CANCELLA FORMULA (CF)**
730 A=VAL(MID$(B$,3,1)):IFA<1ORA>5THENRE
    TURN
740 S$(A)="":RETURN
750 REM **GOTO (GO) ESEMPIO: GO10,2**
760 GOSUB770:R=A1:C=A2:R1=A3:C1=A4:GOSUB
    140:RETURN
770 REM **VALUTAZIONE COMANDO**
780 FORI=3TOLEN(B$):IFMID$(B$,I,1)<>","T
    HENNEXT:GOTO640
790 A=I-3:B=LEN(B$)-I
800 A2=VAL(RIGHT$(B$,B)):A1=VAL(MID$(B$,
    3,A)):IFA1>MRTHENA1=MR
810 IFA2>MCTHENA2=MC
820 A3=INT((A1-.1)/20)*20+1:A4=INT((A2-1
    )/3)+1+2*INT((A2-1)/3):RETURN
830 REM **COPIA COLONNA (CC) ESEMPIO: CC
    2**
840 GOSUB770:GOSUB30:FORI=1TOMR:C$(I,A2)
    =C$(I,A1):NEXT:GOSUB140:RETURN
850 REM **COPIA RIGA (CR) ESEMPIO: CR2**
860 GOSUB770:GOSUB30:FORJ=1TOMC:C$(A2,J)
    =C$(A1,J):NEXT:GOSUB140:RETURN
870 REM **AZZERA SPREADSHEET (ZR)**
880 GOSUB30:FORI=1TOMR:FORJ=1TOMC:C$(I,J
    )="":NEXT:NEXT:GOSUB710:RETURN
890 REM **SAVE (S:)**
900 COLOR4,5:F$=RIGHT$(B$,LEN(B$)-2):IFD
    V=1THENOPEN1,1,1,F$:GOTO920
910 OPEN1,8,3,"@0:"+F$+",S,W"
920 GOSUB30:FORI=1TOMR:FORJ=1TOMC:IFC$(I
    ,J)=""THENPRINT#1,CHR$(160):ELSEPRIN
    T#1,C$(I,J)
930 NEXT:NEXT:CLOSE1:COLOR4,1:SCNCLR:GOS
    UB60:GOSUB710:RETURN
```

```
940 REM **LOAD (L:)**
950 COLOR4,4:F$=RIGHT$(B$,LEN(B$)-2):IFD
    V=1THENOPEN1,1,0,F$:GOTO970
960 OPEN1,8,3,F$
970 GOSUB30:FORI=1TOMR:FORJ=1TOMC:INPUT#
    1,C$(I,J):IFC$(I,J)=CHR$(160)THENC$(
    I,J)=""
980 NEXT:NEXT:CLOSE1:COLOR4,1:SCNCLR:GOS
    UB60:GOSUB710:RETURN
990 REM **COPIA CELLA (CP) ESEMPIO: CP01
    051002**
1000 IFLEN(B$)<>10THEN640
1010 A1=VAL(MID$(B$,3,2)):A2=VAL(MID$(B$
     ,5,2)):A3=VAL(MID$(B$,7,2)):A4=VAL(
     MID$(B$,9,2))
1020 C$(A3,A4)=C$(A1,A2):GOSUB140:RETURN
1030 REM **FORMULE TASTI AMMESSI: +-*/@
     **
1040 IFW$="@"THENW$=""
1050 A$=STR$(R):A$=RIGHT$(A$,LEN(A$)-1):
     IFR<10THENA$="0"+A$
1060 A1$=STR$(C):A1$=RIGHT$(A1$,LEN(A1$)
     -1):IFC<10THENA1$="0"+A1$
1070 S$(Z)=S$(Z)+A$+","+A1$+W$
1080 CHAR1,0,23,"{RVS ON}"+S$(Z)+"{RVS O
     FF}"
1090 IFW$=""THENGOSUB1170:Z=Z+1:IFZ>5THE
     NZ=1:S$(Z)=""
1100 RETURN
1110 REM **VISUALIZZAZIONE FORMULE (VF)*
     *
1120 SCNCLR:PRINT"{HOME}{SH F}{SH O}{SH
     R}{SH M}{SH U}{SH L}{SH E} {SH I}{S
     H N} {SH M}{SH E}{SH M}{SH O}{SH R}
     {SH I}{SH A}:{CUR.GIU}"
1130 FORI=1TO5:PRINTI;" {RVS ON}"S$(I)"{
     RVS OFF}";:IFS$(I)=""THENPRINT"{SH
     N}ON DEFINITA"
```

```
1140 PRINT:NEXT:PRINT"{CUR.GIU}{SH R}{SH
      E}{SH T}{SH U}{SH R}{SH N} RIENTRA
      NELLO SPREADSHEET"
1150 GETKEYA$:IFA$<>CHR$(13)THEN1150
1160 SCNCLR:GOSUB70:GOSUB710:RETURN
1170 REM **ESECUZIONE FORMULE**
1180 B1=VAL(MID$(S$(Z),1,2)):B2=VAL(MID$
     (S$(Z),4,2)):W=VAL(C$(B1,B2))
1190 A=LEN(S$(Z)):FORI=7TOASTEP6
1200 B1=VAL(MID$(S$(Z),I,2)):B2=VAL(MID$
     (S$(Z),I+3,2))
1210 O$=MID$(S$(Z),I-1,1)
1220 IFB1=0ORB2=0THEN1240
1230 GOSUB1250
1240 NEXT:GOSUB710:RETURN
1250 IFO$="+"THENW=W+VAL(C$(B1,B2))
1260 IFO$="-"THENW=W-VAL(C$(B1,B2))
1270 IFO$="*"THENW=W*VAL(C$(B1,B2))
1280 IFO$="/"THENW=W/VAL(C$(B1,B2))
1290 IFO$="="THENC$(B1,B2)=RIGHT$(STR$(W
     ),LEN(STR$(W))-1)
1300 RETURN
1310 REM **FORMULE/RIGHE (FR) ESEMPIO: F
     R2**
1320 B=VAL(MID$(B$,3,1)):IFB<1ORB>5THENR
     ETURN
1330 GOSUB30:FORK=1TOMR:B1=K:B2=VAL(MID$
     (S$(B),4,2)):W=VAL(C$(B1,B2))
1340 A=LEN(S$(B)):FORI=7TOASTEP6
1350 B1=K:B2=VAL(MID$(S$(B),I+3,2))
1360 O$=MID$(S$(B),I-1,1)
1370 IFB2=0THEN1240
1380 GOSUB1250
1390 NEXT:NEXT:GOSUB710:RETURN
1400 REM **FORMULE/COLONNE (FC) ESEMPIO:
      FC2**
1410 B=VAL(MID$(B$,3,1)):IFB<1ORB>5THENR
     ETURN
```

```
1420 GOSUB30:FORK=1TOMC:B1=VAL(MID$(S$(B
     ),1,2)):B2=K:W=VAL(C$(B1,B2))
1430 A=LEN(S$(B)):FORI=7TOASTEP6
1440 B1=VAL(MID$(S$(B),I,2)):B2=K
1450 O$=MID$(S$(B),I-1,1)
1460 IFB1=0THEN1240
1470 GOSUB1250
1480 NEXT:NEXT:GOSUB710:RETURN
1490 REM **MEMORIA LIBERA (ME)**
1500 I=FRE(0)-(FRE(0)<0)*65536:CHAR1,25,
     22,STR$(I)+" BYTE{2 SPC}":FORI=1TO1
     000:NEXT:RETURN
1510 REM **SOMMA COLONNA (SC)**
1520 K=VAL(MID$(B$,3,1)):IFK<1ORK>MCTHEN
     RETURN
1530 A=0:GOSUB30:FORI=1TOMR:A=A+VAL(C$(I
     ,K)):NEXT:C$(MR,K)=STR$(A):RETURN
1540 REM **SOMMA RIGA (SR)**
1550 K=VAL(MID$(B$,3,2)):IFK<1ORK>MRTHEN
     RETURN
1560 A=0:GOSUB30:FORI=1TOMC:A=A+VAL(C$(K
     ,I)):NEXT:C$(K,MC)=STR$(A):RETURN
1570 REM **STAMPA SPREADSHEET (SS)**
1580 GOSUB30:OPEN4,4,7:SP$="{19 SPC}":FO
     RA=1TO7STEP3
1590 PRINT#4,"{10 SPC}{SH C}OLONNA{4 SPC
     }";:FORJ=ATOA+2:PRINT#4,J;"{11 SPC}
     ";:NEXT:PRINT#4
1600 FORI=1TOMR:PRINT#4,LEFT$("{SH R}: "
     +STR$(I)+SP$,10);
1610 FORK=ATOA+2:PRINT#4,RIGHT$(SP$+C$(I
     ,K),14);:NEXT:PRINT#4
1620 NEXT:PRINT#4:PRINT#4:NEXT:CLOSE4:RE
     TURN
1630 REM **INTERCETTAZIONE ERRORI**
1640 SCNCLR:PRINT"{SH A}{2 SH T}{SH E}{S
     H N}{SH Z}{SH I}{SH O}{SH N}{SH E}!
      {SH S}{SH I} {SH E}' {SH V}{SH E}{
     SH R}{SH I}{SH F}{SH I}{SH C}{SH A}
```

```
     {SH T}{SH O} {SH U}{SH N}":PRINT"{S
     H E}{2 SH R}{SH O}{SH R}{SH E} {SH
     D}{SH E}{SH L} {SH T}{SH I}{SH P}{S
     O}:{CUR.GIU}"
1650 PRINTERR$(ER):PRINT
1660 IFER<>20THEN1690
1670 PRINT"{SH O}CCORRE PRESTARE MAGGIOR
     E ATTENZIONE"
1680 PRINT"NELL'EFFETTUAZIONE DEI CALCOL
     I!":GOTO1740
1690 IFER<>18THEN1710
1700 PRINT"{SH U}N VALORE TROPPO ELEVATO
      E' STATO":PRINT"ASSEGNATO A UNA RI
     GA O COLONNA!"
1710 IFER<>16THEN1740
1720 PRINT"{SH O}CCORRE DIMINUIRE IL NUM
     ERO DI CARATTERICONTENUTO NELLA MAG
     GIOR PARTE DELLE"
1730 PRINT"CELLE, FINO A CHE LA MEMORIA
     LIBERA SIA SUFFICIENTE (USARE IL CO
     MANDO 'ME')"
1740 PRINT"{2 CUR.GIU}{SH P}{SH R}{SH E}
     {SH M}{SH I} {SH U}{SH N}{SH SPC}{S
     H T}{SH A}{SH S}{SH T}{SH O} {SH P}
     {SH E}{SH R} {SH R}{SH I}{SH E}{SH
     N}{SH T}{SH R}{SH A}{SH R}{SH E}"
1750 GETKEYA$:SCNCLR:GOSUB70:GOSUB710:RE
     SUMENEXT
1760 REM **SALVA FORMULE (SF)**
1770 COLOR4,5:F$=RIGHT$(B$,LEN(B$)¬2):IF
     DV=1THENOPEN1,1,1,F$:GOTO1790
1780 OPEN1,8,3,"@0:"+F$+",S,W"
1790 GOSUB30:FORI=1TO5:IFS$(I)=""THENPRI
     NT#1,CHR$(160):ELSEPRINT#1,S$(I)
1800 NEXT:CLOSE1:COLOR4,1:SCNCLR:GOSUB60
     :GOSUB710:RETURN
1810 REM **CARICA FORMULE (LF)**
1820 Z=1:COLOR4,4:F$=RIGHT$(B$,LEN(B$)¬2
     ):IFDV=1THENOPEN1,1,0,F$:GOTO1840
```

```
1830 OPEN1,8,3,F$
1840 GOSUB30:FORI=1TO5:S$(I)=""
1850 GET#1,A$:IFA$<>CHR$(13)THENS$(I)=S$
     (I)+A$:GOTO1850
1860 Z=Z+1:IFS$(I)=CHR$(160)THENS$(I)=""
     :Z=Z-1
1870 NEXT:CLOSE1:COLOR4,1:SCNCLR:GOSUB60
     :GOSUB710:RETURN
```

La diffusione sempre più capillare del computer ha fatto comprendere a tutti che esso non è soltanto un sofisticato videogioco, ma anche un mezzo atto a migliorare la qualità del proprio lavoro.
In questo libro sono infatti presentati e commentati quattro potentissimi programmi di qualità commerciale — in BASIC ed in linguaggio macchina — da utilizzare a casa o sul lavoro con il proprio Commodore 16, per soddisfare un'infinità di esigenze diverse.
Il Word Processor per sostituire totalmente qualsiasi macchina per scrivere, due veloci Data Base per archiviare qualunque tipo di informazioni, ed il versatile Spreadsheet per effettuare calcoli complessi ed analisi di mercato, vi permetteranno finalmente di mettere a frutto in modo intelligente l'investimento fatto con l'acquisto del vostro computer.

L. 20.000

Cod. CC244 ISBN 88-7056-316-2

GRUPPO EDITORIALE
JACKSON 244

Lavoriamo con il COMMODORE 16

Alessandro Borra   Mauro Cristuib Grizzi